MIT Libraries
Document Services

Room 14-0551
77 Massachusetts Avenue
Cambridge, MA 02139
Ph: 617.253.5668 Fax: 617.253.1690
Email: docs@mit.edu
http://libraries.mit.edu/docs

DISCLAIMER OF QUALITY

Due to the condition of the original material, there are unavoidable flaws in this reproduction. We have made every effort possible to provide you with the best copy available. If you are dissatisfied with this product and find it unusable, please contact Document Services as soon as possible.

Thank you.

Appendix 1 contains poor quality text.

MASS. INST. OF TECHNOLOGY
JUN 27 1962
LIBRARY

A CHESS PLAYING PROGRAM FOR THE IBM 7090 COMPUTER

by

Alan Kotok

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF BACHELOR OF SCIENCE

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June, 1962

Signature of Author . . . . . .
Department of Electrical Engineering

Certified by . . . . . .
Thesis Supervisor

Accepted by . . . . . .
Chairman, Departmental Senior Thesis Committee

ABSTRACT

This paper covers the development of a chess playing program. The preliminary planning led to the decision to use a variable depth search, terminating at either an arbitrary maximum, or at a stable position. Two schemes of controlling material balance are discussed. Of major significance is the use of the "alpha-beta" heuristic, a method of pruning the tree of moves. This heuristic makes use of values obtained at previous branches in the tree to eliminate the necessity to search obviously worse branches later.

The program has played four long game fragments in which it played chess comparable to an amateur with about 100 games experience.

ACKNOWLEDGMENT

I wish to thank Michael Lieberman, Charles Niessen, and Robert Wagner, the current members of the MIT chess group, for their invaluable assistance in this project. I also wish to express my appreciation to Elwyn Berlekamp, B. F. Wells and Paul Abrahams, who were previously associated with this project.

Special thanks go to Prof. John McCarthy who has guided the chess program through good days and bad. I wish to acknowledge the cooperation of the MIT Computation Center for providing the computation facilities necessary for this project.

Lastly, I wish to thank Robert Saunders for his help with the programming, and Milton Garber and Robert Fiorenza for giving their time to play against the machine.

TABLE of CONTENTS

Abstract . . . . . -ii-

Acknowledgment . . . . . -iii-

Introduction . . . . . -1-

Preliminary Investigation. . . . . -1-

Organization of the Chess Program. . . . . -4-

Description of Component Sub-Programs. . . . . -7-

Administrative Routines. . . . . -7-

Plausible Move Generation. . . . . -8-

Evaluation Routines. . . . . -8-

Service Routines . . . . . -10-

Input-Output Routines. . . . . -11-

Results. . . . . -12-

Figure 1 - representative output . . . . . -13-

Appendix 1 - Listing of the Chess Program

Appendix 2 - Sample Input to Initia

Appendix 3 - Record of Chess Games

INTRODUCTION

This thesis describes a chess playing program for the IBM 7090 computer. Although chess programs have been previously written, none of these played what could be considered "good chess".

Before commencing work on our chess program, we studied the report published by Newell, Shaw and Simon covering previous attempts, such as the Los Alamos program, and Bernstein's program at IBM.

PRELIMINARY INVESTIGATION

The chess group, consisting of Messrs. Berlekamp, Niessen, Lieberman and Kotok, inherited routines for generating and making legal moves. With these as a basis, we decided to write a three move mate solving program for the purpose of familiarizing ourselves with the existing routines, and to come in contact with many of the problems we would later face in the actual general playing program. The three move mate program was completed in the spring of 1960. It was given problems from actual games, and successfully solved many of them. The three move mate program was written for the IBM 704, which was removed from the MIT Computation Center in the summer of 1960. Due to incompatibility with the incoming 709, the project was dropped at the end of the spring term of 1960.

In the fall of 1960 the chess group, without Mr. Berlekamp, began planning for the general chess program.

It was decided to retain the original McCarthy-Abrahams move routines, and to continue coding in FORTRAN and FAP. The program was to be a variable depth search with a "stable position" termination. An evaluation was to be made at the terminal points of the move tree. This evaluation would be a weighted sum of such criteria as material balance, center control, pawn structure, "tempo" advantage, and development.

Moves on each level were to be proposed by "plausible move generators" which would propose moves to fulfill various goals. As the tree was searched, a backing up process would take place, in which the move declared best at each level by the evaluation would have its value brought up to the next higher level.

This procedure, also called mini-max, leads to a "principal variation" which is that set of moves which the machine considers most likely to happen. The evaluation always assumes that a player will always make the best move available to him a given time.

It was, of course, recognized that any evaluation could not be perfect, since chess is a game in which the only way a position can be perfectly evaluated is to look to the end of the game, and see whether it leads to a win, draw, or loss. The only sound basis for an evaluation is that chess masters have, over the years, accumulated knowledge concerning the play of the game. For instance, a position in which a piece is "en prise" is considered

bad, while having rooks on open files is considered good, even though the rules do not state anything about such things.

Since none of the members of the chess group are more than amateurs, we consulted books by masters to find out how much better it is to control the center than to have a strong pawn structure. These books are amazingly elusive on such details. Although many tips were given concerning the play of the game, relative importance of various strategies was uncertain.

We therefore considered having the program play for a while, and adjust the weights of the evaluation criteria to optimize its position. Although such a scheme seemed desirable, it was decided not to include any "learning" in the program due to the unavailability of suitably large amounts of computer time.

ORGANIZATION OF THE CHESS PROGRAM

Work on the chess program itself began in the spring term of 1961. The program is written in subroutine form, using the Fortran Monitor System of linkage. Where possible, programs are written in FORTRAN, and where it becomes too clumsy, or inefficient, FAP is used.

The actual implementation of the above mentioned "plausible move generators" has never been accomplished. Instead, we have a program, called REPLYS, which scans the legal move table, updates, evaluates, and reverts each move and orders them according to a single ply evaluation. (A ply is a half-move, i.e. a move by only one side.) The number of moves actually chosen is a function of the current depth in the tree.

Evaluation functions were written for material balance, center control, and development, since we intended to concentrate our efforts on openings until the program was thoroughly debugged.

The coordinating routine written in the spring of 1961, called TREE, employed the above mentioned mini-max scheme. REPLYS was set to cut the search at a depth of eight plys, or whenever the situation was stable, whichever came first.

The program was tested late in the spring of 1961. The 709 took about 5 to 20 minutes per move, depending on the complexity of the situation. Although the machine did not do too badly, we noted that it was looking at many

irrelevant positions. We thefore attempted to find a method of pruning the move tree, without discarding good as well as bad moves.

Prof. McCarthy proposed a heuristic for this purpose, called "alpha-beta". It operates as follows: Alpha is a number representing the value of the best position which white can reach, using a pessimistic evaluation. Beta represents the best position white can reach, using an optimistic evaluation, due to the fact that black can hold him to this position. Under normal circumstances, alpha starts at -infinity, and beta at +infinity.
At each level, optimistic and pessimistic evaluations are made, and compared to alpha and beta in the following way. If a white move is optimistically less than alpha, it is discarded, since a better alternative exists elsewhere. Likewise, if a white move pessimistically is better than beta, it too is discarded, since black had a better alternative previously; furthermore we revert two levels since no other white moves are worth considering at that position. The reverse strategy is applied for black.

The "alpha-beta" version of TREE was written during the summer of 1961, and was first put to use during the fall of that year. Also, we were joined by Mr. Wagner in the fall term of 1961.

After testing in the fall of 1961, it was decided that the material balance programs were insufficient. We therefore decided to replace the scheme then in use with

a new, updated scheme. The programs then in use, and, as it happens, in use now, completely re-generate the material balance function at each position.

The material balance evaluator consists of two subroutines, SWAP and LTRADE. SWAP's function is to list all attacks and defences on each occupied square. Secondary attackers which reside behind primary attackers (or defenders) are included. The pieces are listed in the order in which they would be played. Lowest valued pieces come first, unless the order is disturbed by the necessity of a higher valued piece to move first due to position. Pieces pinned to the king and queen were not recognized, leading to embarrassing evaluations. Likewise, discovered attacks were not considered.

LTRADE then simulates trade-off of all attacked pieces, and chooses the line most profitable for the side to move. The opponent is given the option of having a given piece taken, or moving the piece away. After all possible trades have been made, the program computes whether it is to the advantage of the machine to initiate an exchange, and if so, what the probable gain would be.

This scheme is both time consuming, and occasionally inaccurate. It was therefore decided to write a new evaluator for the material balance, which kept an updated set of tables, in a list structure format, from which the outcome of a given exchange could be found at a glance.

After a few months of planning and programming, the new list structure program was found to be impractical, due to excessive complication in the update procedure. Furthermore, the values which were to be included in the list were found to be no more accurate than the ones which the above scheme produced. The project was therefore abandoned.

DESCRIPTION OF COMPONENT SUB-PROGRAMS

The chess program is organized into a non-recursive hierarchy of sub-programs. Listings are to be found in appendix 1.

ADMINISTRATIVE ROUTINES

(MAIN) This is the highest level program. The on-line main program has the job of handling input-output, and timing. It determines the opponent's move by looking at the console keys, and picks the appropriate move from the legal move table. It then calls TREE which actually makes the move, after which (MAIN) prints out the machine's reply.

TREE Tree is the second level of control. Tree has the responsibility of constructing the tree of legal moves. It calls REPLYS to generate a list of plausible moves, and enters these in the LISP table, which is the actual tree. The moves are then chosen in order of decreasing value, and

updated. A new list of plausible moves is then generated for the opponent. The optimistic and pessimistic evaluators are called, and the alpha-beta tests are made, as described above. In the event that no replies are generated, due to stability, or excessive depth, a static evaluation is made and assigned to the position. The last move is then reverted, and the search proceeds down the next most likely branch of the tree. When all desired positions have been examined, the "best" move is returned as the answer.

PLAUSIBLE MOVE GENERATION

REPLYS This program supplies lists of plausible moves to TREE. It updates each of the legal moves, evaluates the position and reverts. The number of moves presented is a function of the present ply. Current values in order of increasing ply are: 4 3 2 2 1 1 1 1 0 0. These are input parameters to the program.

EVALUATION ROUTINES

EVAL Eval is the static evaluation program. Its function is to call all the subsidiary evaluation programs and to apply suitable multipliers, and hence form a weighted sum. Material values are: pawn 1, knight and bishop 3, rook 5, queen 9, and king 1000. These values are normally multiplied by 60 when combined with the other functions. Should one side be ahead at least 4 points, the material multipliers are adjusted to make trading

off advantageous.

LTRADE This program, described in more detail above, provides the projected material gain, considering all attacks and defenses.

ICENTR The center control evaluator gives points for controlling the 16 center squares. Looking from either side, these values are:

8	8	4	4
4	8	8	4
2	4	4	2
1	1	1	1

The center control points are each worth 1/60 of a pawn. After the game passes the twentieth full move, the center control function is decreased in importance until the 30th move, when it is discarded.

IDVLOP The development function, gives points for each developed piece. These range from 1 point per pawn, to 3 or 4 points for other pieces. Development points are weighted 1/15 of material points. This function is also eliminated as the game progresses.

JPAWNS The pawn structure function, considers the following situations, with aproximate point values:

open file	+8

isolated pawn	-1
backward pawn	-5
doubled pawn	-3
passed pawn	+10

These points are weighted 1/20 of material points.

SERVICE ROUTINES

UPDATE Updates any legal move, and records all relevant information on a push-down list. It then generates all legal replies available to the other side, using the general purpose move routines UPREV and PUTCH.

REVERT Takes back the last updated move. This is actually another option of the the updating routine UPREV.

PUTCH A lower level routine used in making moves. It keeps tables of almost legal moves and piece bearings updated. This table does not include castling, and "en passant" moves.

SWAP Generates the list of all attacks and defenses on occupied squares, listed in the order in which the pieces would be played.

PINS Generates the list of all pieces pinned to Kings and Queens. Includes the pinning direction, so that SWAP will only consider a pinned piece as an attacker or defend-

er along the line of the pin.

INPUT-OUTPUT ROUTINES

PRINT The major output routine. It handles most of the printing, both on and off line. It, and its subroutines, print the chess board, legal move table, principal variation, move tree and log of all moves tried, plus other information useful in debugging.

INITIA Reads in any chess board position. Its input language is as follows.

The chess board is scanned, from left to right, starting at white's Queen Rook 1. Digits represent numbers of unoccupied squares. Pieces are represented by the normal chess notation, in its most explicit form; e.g. KBP for King Bishop Pawn. Black pieces are preceded by asterisks. After exactly 64 squares are specified, the character"." (period) signifies the end of the specification and that white is to move. "*." indicates black to move. Additional features include the ability to indicate promoted pawns, by stating the type of piece, followed by the name of the pawn from which it promoted, in parentheses; e.g. Q(KNP). Also, it is possible to indicate that a piece has previously moved (for rooks, kings and pawns) by suffixing (M) to the piece name. Comments must begin and end with slashes.

The input is on IBM cards, punched in columns 1

through 72, taking as many cards as necessary. In case of errors found by INITIA, a comment will be printed, the remaining part of the problem will be skipped, and the next problem will be used.

All tables are initialized, and the program is set to commence with the legal move table generated for the side indicated. An example of an INITIA input will be found in Appendix 2.

RESULTS

As of this date, the machine has not completed any chess games. We have, however, played 4 lenghty fragments of games, and also have investigated many individual positions.

For our first long machine run, we chose an undergraduate student, Milton Garber, who held second place in his dormitory chess tournament. A record of this, and other game fragments is to be found in Appendix 3.

The second game was also played against Mr. Garber. In the record of this game a column indicating the principal variation is included. These are the moves the machine considers most likely to happen in succeeding plays, based on the evaluation and minimax process.

In seventeen moves, the machine guessed correctly only thrice, including only one case where it predicted correctly more than one move ahead.

Figure 1 consists of a set of representative

FIGURE 1

Representative output

```
SET OF TABLES NUMBER  4  MOVE IS *QP - Q4
                    BLACK
                    -----
*****************************************
* QR * QN * QB *  Q *  K * KB * KN * KR *
* ---* ---* ---* ---* ---* ---* ---* ---*
*****************************************
* QRP* QNP* QBP*    * KP * KBP* KNP* KRP*
* ---* ---* ---*    * ---* ---* ---* ---*
*****************************************
*    *    *    *    *    *    *    *    *
*    *    *    *    *    *    *    *    *
*****************************************
*    *    *    * QP *    *    *    *    *
*    *    *    * ---*    *    *    *    *
*****************************************
*    *    *    * QP *    *    *    *    *
*    *    *    *    *    *    *    *    *
*****************************************
*    *    *    *    *    *    *    *    *
*    *    *    *    *    *    *    *    *
*****************************************
* QRP* QNP* QBP*    * KP * KBP* KNP* KRP*
*    *    *    *    *    *    *    *    *
*****************************************
* QR * QN * QB *  Q *  K * KB * KN * KR *
*    *    *    *    *    *    *    *    *
*****************************************
                    WHITE
MAVAIL

  K - Q2      QRP-QR3     QRP-QR4     QNP-QN3     QNP-QN4     QBP-QB3     QBP-QB4     KP - K3     KP - K4     KBP-KB3

KBP-KB4     KNP-KN3     KNP-KN4     KRP-KR3     KRP-KR4     QN - Q2     QN -QB3     QN -QR3     KN -KR3     KN -KB3

QB - Q2     QB - K3     QB -KB4     QB -KN5     QB -KR6      Q - Q2      Q - Q3
```

```
PRINCIPAL VARIATION
VALUE=     27 EFFORT=    1449
*QP - Q4      KN -KB3
```

Figure 1 (cont)

THE MOVE TREE

LEVEL 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	VALUE
*QP - Q4																				
	QN -QB3																			27
	KN -KB3																			27
	QB -KB4																			
		*QN -QB3																		
			QN -QB3																	
				*KN -KB3																
					KN -KB3															
						*QB -KB4														
							QB - K5													21
		KN -KB3																		
			*KN -KB3																	
				QN -QB3																
					*QB -KB4															
						QB - K5														21
	*KN -KB3																			
*QN -QB3																				
QP - Q5																				
	*QN -QN5																			
		KP - K4																		58
		QN -QB3																		54
	*QN -QR4																			
		KP - K4																		66
		QN -QB3																		
KP - K4																				
QN -QB3																				
*KN -KB3																				
QN -QB3																				33
KN -KB3																				
QB -KB4																				
*QP - Q3																				
KP - K4																				46
QN -QB3																				
KN -KB3																				

Fig. 1

(cont)

output for a single move. The first page is a printout of the chess board, and a list of the opponents legal replies, labeled MAVAIL. The second page contains the principal variation, beginning with the value of this variation, and the number of positions examined at the approximate rate of 1100 positions per minute. The principal variation itself commences with the machine's move.

The following pages contain the actual move tree. The moves listed therein are moves which were considered plausible by the reply generator. Moves were considered in the order top to bottom, however all moves on level one were generated simultaneously, and all level two replies to each level one move are generated together, etc. The "value" column contains a value on each terminating position. Values of $\pm$131071 indicate positions discarded for alpha-beta cutoff. Terminating positions which have no values have not even been examined, since the alpha-beta heuristic found previous moves on that level to be either too good, or too bad.

A third game fragment was played against an amateur with little chess experience; in particular, he knew the game, and had played some before he came to MIT. The game progressed 34 moves before time expired, with the result that the machine was ahead 1 rook, 2 knights and 2 bishops.

From our analysis of the results, we have found that in its present state, the program is comparable to

an amatuer with about 100 games experience.

Most of the machine's moves are neither brilliant nor stupid. It must be admitted that it occasionally blunders. These blunders can often be traced to wrong multiplyers in the evaluation, and occasionally to situations where discovered attacks, forks, etc. cause confusion. It is rare, however, not to find the correct move in the list of plausible moves.

This study is far from complete, but we feel that our efforts are proving fruitful. Hopefully this work will be continued.

APPENDIX 1

```
*       LABEL
*       FAP
        COUNT   400
*TREE    FUNCTION FOR CHESS WITH ERROR PRINT, MAR. 2, 1962
*
*
*       GIVEN A MOVE AS THE FIRST ARG, IT GENERATES A TREE OF MOVES,
*       MINIMAXES, AND ITS VALUE IS THE DESIRED REPLY IN ,MOVE, FORMAT.
*       THE FORMAT OF THE TABLE IT GENERATES (CALLED LISP) IS AS FOLLOWS-
*
*                                       MOVE        BACK
*                                       VALUE  PLY     N
*                                       REPLY(1) POINTER(1)
*                                       REPLY(2) POINTER(2)
*                                       .    .    .    .    .
*                                       -REPLY(N) POINTER(N)
*
*       THE ABOVE IS 1 BLOCK IN THE LISP TABLE. IT IS GENERATED ONLY ONCE
*       MOVE IS THE MOVE UNDER CONSIDERATION, IN BITS 3-20. THE SIGN MAY
*       BE NEGATIVE IF THERE ARE NO PROPOSED REPLIES.
*       BACK IS THE INDEX OF THE FIRST WORD OF THE BLOCK FROM WHENCE
*       WE CAME.  (NOTE-  ALL SUCH INDICES MAY BE OFF BY A CONSTANT.)
*       VALUE IS THE VALUE OF THE MOVE AS DETERMINED BY MINIMAXING.
*       N IS THE NUMBER OF REPLIES NOT YET CONSIDERED, WHICH IS COUNTED
*       DOWN TO ZERO, AT WHICH TIME THE MOVE IS EVALUATED, AND N BECOMES
*       THE INDEX OF THE REPLY THAT LED TO THE VALUE CHOSEN.
*       SINCE THE ABOVE EXPLANATION IS SO CLEAR, COMMENTS WILL NOT BE
*       PROVIDED ADJACENT TO THE PROGRAM, SINCE THESE WILL ONLY SERVE TO
*       ADD TO THE ALREADY ABUNDANT CONFUSION.  SO HERE IT IS.....
*       DIMENSION IHOPE(64), LISP(6000)
*       NEXT FREE REGISTER IN COMMON = 23375
*
*               INITIALIZE
        ENTRY   TREE
TREE    SXA     XR1,1
        SXA     XR1+1,2
        SXD     XR4,4
XR4     SYN     TREE-2
        AXT     3000,1
        STZ     LISP+1,1
        TIX     *-1,1,1
        STZ     MOVE
        CALL    STRTGY
        AXT     1,1
        STZ     BACK
        STZ     PLY
*
*               GENERATE A NEW BLOCK.
 D      CLA     PLY
        ADD     =0200
        STO     PLY
        CLA     MOVE            HEAD NEW BLOCK
        ADD     BACK
        SXA     BACK,1
        STO     LISP+1,1
```

```
        CLA     PLY
        ANA     =0200
        TNZ     OUT
APB     LXD     MCOL,4      BEGIN COMPARISON OF A,B 2 BLOCKS HIGHER
        CLA     LISP+1,1
        PAX     ,2
        TXL     FO1,2,0
        CLA     LISP+1,2
        PAX     ,2
        TXL     FO1,2,0
        CLA     LISP-1,2
        TPL     *+3
        CLA     LISP+1,2
        TRA     APB+2
MN      XEC     SPG+1,4
        TXI     *+2,0,
        PZE     TREE-2,0,MN
        STO     VALUE
        LDI     =1
        STI     ID
        LXD     MCOL,4
        CAS     LISP,2
        XEC     TS1+1,4
        TRA     *+2
        XEC     TS1+2,4
        SLW     A
        ADD     PLY      FAIL TEST 1-- REVERT TWICE.(PASS, TRA FO1)
        SLW     LISP,1
        CLA     LISP+1,1
FF      PAX     ,2
        CAL     LISP,2
        ANA     =0777777
        ORA     A
        ADD     =1
        SLW     LISP,2
        CALL    REVERT
        CLA     PLY
        SUB     =0400
        STO     PLY
        CLA     LISP+1,2
        PAX     ,2
        CALL    REVERT
        CLA     LISP-1,2
        TMI     GG     (SINGLE REPLY CHAIN--GO BACK 2 MORE LEVELS)
DN      SXA     BACK,2
        CAL     =-0
        ORS     LISP+1,1
        SXA     RX4,4
        TSX     PT,4
        LXA     RX4,4
        TXI     C,1,2
*
GG      CAL     LISP,2
        ANA     =0777777
```

```
        ORA     A
        ADD     =1
        SLW     LISP,2
        CLA     LISP+1,2
        TRA     FF
*
        TSX     $PESVL,4    FUNCTIONS RETURN IN ALGEBRAIC FORM
 SPG    TSX     $OPTVL,4
        TSX     $PESVL,4
        CAL     =0377777000000
 TS1    TRA     FO1
        CAL     =0777777000000
 TS2    TRA     OUT
        CAL     =0377777000000
*
 FO1    CLA     LISP+1,1    BEGIN TEST 2--A,B ONE BLOCK HIGHER
        PAX     ,2
        TXL     OUT,2,0
        CLA     LISP-1,2
        TPL     *+6
        CLA     LISP+1,2
        PAX     ,2
        TXL     OUT,2,0
        CLA     LISP+1,2
        TRA     FO1+1
 NM     XEC     SPG+2,4
        TXI     *+2,0,0
        PZE     TREE-2,0,NM
        STO     VALUE
        LDI     =2
        STI     ID
        LXD     MCOL,4
        CAS     LISP,2
        XEC     TS2+2,4
        TRA     *+2
        XEC     TS2+1,4
        SLW     A
        ADD     PLY         FAIL TEST 2 REVERT    (PASS, TRA OUT)
        SLW     LISP,1
        CLA     LISP+1,1
 FG     PAX     ,2
        CALL    REVERT
        CLA     LISP-1,2
        TMI     GF      SINGLE REPLY CHAIL--GO BACK 2 MORE LEVELS
        CLA     PLY
        SUB     =0200
        STO     PLY
        TRA     DN
 GF     CAL     LISP,2
        ANA     =0777777
        ORA     A
        ADD     =1
        SLW     LISP,2
        CLA     PLY
```

```
         SUB     =0400
         STO     PLY
         CLA     LISP+1,2
         PAX     ,2
         CAL     LISP,2
         ANA     =0777777
         ORA     A
         ADD     =1
         SLW     LISP,2
         CALL    REVERT
         CLA     LISP+1,2
         TRA     FG
*
         ORA     =0377777000000
  INT    ORA     =0777777000000
  OUT    CALL    REPLYS
         LXD     IPE,2
         PXA     ,2
         ADD     PLY
         SLW     LISP,1
         TXL     B,2,0     (NO REPLYS--POSITION STATIC)
         LXD     MCOL,4
         XEC     INT+1,4
         SLW     LISP,1
         SXD     AF,2
         AXT     1,2
Q        CLA     IHOPE+1,2          LIST PRODUCED BY REPLYS
         STO     LISP-1,1
         TXI     *+1,1,1
         TXI     *+1,2,1
AF       TXL     Q,2,**
         TXH     ERR,1,3000
         CAL     =-0
         ORS     LISP,1
         TXI     C,1,2
B        CAL     =-0
         ORS     LISP+1,1
         TXI     NOMOVE,1,2
*
ERR      CALL    ERROR,FMT
         TSX     $LDUMP,4
+MT      BCI     2,LISP FULL.
         SVN     -1,7,-1
*                UPDATE THE NEXT REPLY WITHIN A BLOCK.
C        LXA     BACK,2
         CLA     LISP,2
         ANA     =0177
         TZE     USEDUP       (ALL REPLIES USED UP)
         ADD     BACK
         PAX     ,4
         SXA     G,4
         CLA     LISP,4
         STO     RMOVE
         SLW     MOVE
```

```
         CALL    UPDATE,MOVE
         CLA     RMOVE      SHIFT PROMOTION INFORMATION
         LRS     0
         STD     AA
         ALS     15
         ANA     =0700000
         ADD     AA
         LLS     0
G        AXT     **,4
         STO     LISP,4
         SLW     MOVE
         PXA     ,1
         STA     LISP,4
         CAL     LISP,2
         SUB     =1
         SLW     LISP,2
         TRA     D
*
*                THERE ARE NO ENTRIES IN IHOPE.  EVALUATE THE QUOTE
*                STATIC  UNQUOTE POSITION.
NOMOVE   CLA     LISP+3,1
         STA     BACK
         CALL    EVAL
         ORS     LISP+2,1     VALUE RETURNED IN LOGICAL FORM
         CLA     LISP+2,1
LK       LXD     MCOL,4
         LXA     BACK,2
         CAS     LISP,2   MINIMAX VALUE INTO NEST HIGHER LEVEL
         XEC     BRN+1,4
         TRA     *+2
         XEC     BRN+2,4
         STO     A
         CAL     A
         STP     LISP,2
         CLA     PLY      CHANGE NPLY
         ARS     7
         PAX     ,4
         CLA     LISP,2
         ANA     =0177
         ADD     =01
         STO     NPLY+2,4
         CLA     LISP+1,2            A,B TEST
         STA     A
NIN      LXA     A,4
         TXL     OT,4,0
         CLA     LISP-1,4
         TPL     IN
         CLA     LISP+1,4
         PAX     ,4
         TXL     OT,4,0
         CLA     LISP+1,4
         STA     A
         TRA     NIN
IN       CLA     LISP,2
```

```
         LXD     MCOL,4
         LXA     A,2
         CAS     LISP,2
         XEC     BNR+1,4
         TRA     *+2
         XEC     BNR+2,4
         LXA     BACK,2        PASS TEST  (FAIL, TRA OT)
 GRA     CAL     LISP,2
         ADD     =1
         SLW     LISP,2
         CLA     LISP+1,2
         PAX     ,4
         CLA     LISP-1,4
         TMI     ARG     (SINGLE REPLY CHAIN)
         SXA     BACK,4
         CLA     PLY
         SUB     =0400
         STO     PLY
         CALL    REVERT
         CALL    REVERT
         TRA     C
*
         STO     TEST
 BNR     TRA     OT
         STO     TEST
*
 ARG     CLA     LISP+1,4
         PAX     ,2
         LDQ     TEST
         SLQ     LISP,4
         SLQ     LISP,2
         CAL     LISP,4
         ADD     =1
         SLW     LISP,4
         CLA     PLY
         SUB     =0400
         STO     PLY
         CALL    REVERT
         CALL    REVERT
         TRA     GRA
 OT      CLA     PLY
         SUB     =0200
         STO     PLY
         CALL    REVERT
         TRA     C
         STD     LISP,2
 BRN     TRA     OT
         STD     LISP,2
*
*
USEDUP   CLA     LISP+1,2   ALL REPLYS IN BLOCK USED
         STA     BACK
         CLA     PLY
         ARS     7
```

```
         PAX     ,4       FIX NPLY
         CLA     NPLY+1,4
         ORS     LISP,2
         CLA     PLY
         SUB     =0200
         TZE     DONE
         CLA     LISP,2
         TRA     LK
*
*
*                EXIT GLEEFULLY WITH THE BEST MOVE IN THE AC.
DONE     CLA     LISP-1
         ANA     =0177
         SXA     SHMACK,1
         PAX     ,2
         CLA     LISP-1,2                 REPLACE PROMOTION INFORMATION
         STD     AA
         ANA     =0700000
         ARS     15
         ADD     AA
         SXA     RX4,4
         TSX     PT11,4
         CLA     AC
         LXA     RX4,4
XR1      AXT     **,1
         AXT     **,2
         LXD     XR4,4
         STO*    1,4
         TRA     2,4
MOVE     PZE
BACK     PZE
AA       PZE
 A       PZE
RMOVE    PZE
TEST     PZE
VALUE    PZE
M        PZE
         BSS     19
 NPLY    BSS     1
*
PT       SXA     WW,4
         TSX     PTA,4
TP       LXA     RX1,1
         LXA     RX2,2
         LXA     WW,4
         TRA     1,4
PT11     LDI     =11
         STI     ID
         SXA     WW,4
         TSX     PTA,4
         TSX     PTL,4
         TSX     PTP,4
         TRA     TP
*
```

```
 ID     PZE
 AC     PZE
 WW     PZE
 RX1    PZE
 RX2    PZE
 RX4    PZE
*
 PTA    SXA     RX1,1
        SXA     RX2,2
        SXA     QQ,4
        STO     AC
        ORS     AC
        TSX     $(SPH),4
        PZE     FMTT,,-1
        LDQ     ID
        STR
        LDQ     AC
        STR
        LDQ     LISP,2
        STR
        LDQ     PLY
        STR
        LDQ     MCOL
        STR
        LDQ     BACK
        STR
        LDQ     MOVE
        STR
        LDQ     IPE
        STR
        LDQ     RX1
        STR
        LDQ     RX2
        STR
        LDQ     RX4
        STR
        LDQ     VALUE
        STR
        TSX     $(FIL),4
 QQ     AXT     **,4
        TRA     1,4
FMTT    BCI     6,(14H4THIS IS POINT,05//(6020))
*
 PTL    SXA     BK,4
        CLA     RX1
        ADD     =2
        ALS     18
        STD     EPI
        TSX     $(SPH),4
        PZE     FOR,,-1
        AXT     1,1
RK      PXA     ,1
        XCA
        STR
```

```
        LDQ     LISP+1,1
        STR
        TXI     *+1,1,1
 EPI    TXL     RK,1,**
        TSX     $(FIL),4
 BK     AXT     **,4
        TRA     1,4
*
 PTP    SXA     GB,4
        TSX     $(SPH),4
        PZE     FOR,,-1
        AXT     1,1
SIK     PXA     ,1
        XCA
        STR
        LDQ     NPLY+1,1
        STR
        TXI     *+1,1,1
        TXL     SIK,1,10
        TSX     $(FIL),4
 GB     AXT     **,4
        TRA     1,4
FOR     BCI     3,(//(10X,04:020))
*
*
ZILCH   COMMON  12561
R       COMMON  1
K1      SYN     R+9670
MCOL    SYN     R+9662
IPE     SYN     R+9442
PLY     SYN     R+9441
SHMACK  SYN     R+9440
IHOPE   SYN     R+9439
LISP    SYN     R+9375
        END
```

```
*       LABEL
*        FAP
*SWAP   SOUBROUTINE, FOR MATERIAL BALANCE, 3/5/62
        COUNT   250
*
*       GENERATES THE IEXCH TABLE WHICH CONTAINS,FOR EACH PIECE,  ALL
*       ATTACKERS AND DEFENDERS, LISTED IN ORDER OF USAGE.  THE TABLE
*       IS ARRANGED AS FOLLOWS----
*       ENTRIES 1 THRU 33 CONTAIN INFORMATION ABOUT EACH PIECE.
*       THE DECREMENT CONTAINS THE INDEX OF THE BEGINNING OF ENTRIES
*       IN THE REST OF THE TABLE FOR THAT PIECE, THE END OF SUCH ENTRIES
*       IS THE DEC. OF THE ENTRY OF THE NEXT HIGHER NUMBERED PIECE-1.
*       THE TAG CONTAINS THE NO. OF ATTACKERS AND THE PREFIX HAS THE NO.
*       OF DEFENDERS.  THE ADDRESS CONTAINS THE FIRST USE OF THIS
*       PIECE AS AN ATTACKER OR DEFENDER.  THIS WILL BE ZERO IF NOT USED.
*       THE REST OF THE TABLE CONTAINS THE LIST OF ATT. AND DEFS.
*       THE DEC. OF A WORD WILL CONTAIN THE ATT. OR DEF. PIECE NUMBER.
*       THE TAG CONTAINS (IF THE SIGN BIT IS 1) THE INDEX RELATIVE TO
*       THE BEGINNING OF THIS PARTICULAR SET OF ENTRIES OF THE PIECE
*       WHICH MUST MOVE FIRST DUE TO MASKING.   THE ADDRESS CONTAINS
*       MORE OF THE CHAIN OF USES OF THIS PIECE.
*       THE ADDRESS WILL BE ZERO IF THIS IS THE LAST USE ON THE CHAIN.
*
        ENTRY   SWAP
SWAP    LDQ     =0707070707070      MAKE MQ LOOK PRETTY
        SXA     XR1,1               SAVE XRS
        SXA     XR1+1,2
        SXD     XR4,4
XR4     SYN     SWAP-2
        STI     INDIC               SAVE INDICATORS
        AXT     33,1            INITIALIZE CHAIN AND IECCH
        PXA     ,1
        STO     CHAIN+1,1
        STZ     IEXCH+1,1
        TIX     *-3,1,1
        AXT     34,1                SET COUNT ON IEXCH TO 34
        SXD     COUNT,1
        AXT     1,1
A       SXD     K,1                 MAJOR PIECE LOOP RETURN -OIDT
        CLA     LOC+1,1             IS PIECE ON BOARD
        STZ     ATACK
        TZE     Y                   NO
        SUB     =1B17               YES. SET XR2
        PDX     ,2              XR2 HAS LOC(PIECE)-1
        AXT     960,4           XR4 COUNTS DIRECTIONS BY SIXTEENS
        AXT     0,1                 XR1 INDEXES INTER
        ZET     IBEAR,6             IS IBEAR(SQUARE, DIRECTION) = 0
        TXI     C,1,1               NO, BEARING PIECE IS FOUND
D       TIX     *-2,4,64        YES, DECREMENT DIRECTION
        TXL     D1,4,0                   CORRECT HACK  NUMBER 1
        ZET     IBEAR,2             GET LAST DIRECTION
        TXI     D-1,4,-64       CORRECT HACK NUMBERS 1 AND 2
D1      CLA     COUNT                    CORRECT HACK  NUMBER 1
        TXH     ORDER,1,0           TRA IF BEARERS FOUND
```

```
 E      SXD     ATACK,1
        LXD     K,1
        STD     IEXCH+1,1       SET BEG FOR UNATTACKED PIECES
        CLA     ATACK
        ALS     15
        STP     IEXCH+1,1
        TXI     *+1,1,1
 W      TXL     A,1,32                  CLOSE OF MAJOR PIECE LOOP
        CLA     COUNT
        STO     IEXCH-32                SET LAST BEG (REALLY END FOR PC 32)
        AXT     32,2
 Z      CLA     CHAIN+1,2               ZERO ADDRESSES OF PIECES NOT USED
        PAX     ,1
        PXD
        STA     IEXCH+1,1
        TIX     Z,2,1
 XR1    AXT     **,1                    RESTORE INDEX REGISTERS
        AXT     **,2
        LXD     XR4,4
        LDI     INDIC
        TRA     1,4                     RETURN
 *
 X      TXI     D,1,-1                  USED FOR VERTICAL PAWNS
 Y      CLA     COUNT                   USED FOR PIECE OFF BOARD
        TRA     E+1
 *      PHASE 1, SET UP INTER WITH ALL BEARERS IN RANDOM ORDER
  C     CLA     IBEAR,6                 PICK UP BEARER
        TMI     X                       TRA IF VERT PAWN
        SXA     F,2                     SAVE SQUARE
        PDX     ,2                      PIECE TO XR2
        PAI
        ZET     KPIN+1,2                IS THIS PIECE PINNED
        TRA     PIND                    YES
 C1     STO     INTER+1,1               ENTER BEARER IN INTR
        IIS     K
        LFT     1
        STL     ATACK                   YES. SET FLIP-FLOP
 H      CLA     LOC+1,2                 NO
        SUB     =1B17
        PDX     ,2              XR2 HAS LOC(BEARER)-1
        CAL     IBEAR,6                 DO WE HAVE MASKED PIECE
        TNZ     G                       YES
 F      AXT     **,2                    NO, RESTORE XR2 TO ORIGINAL PIECE
        TXL     D,1,19                  AND EXIT
        TRA     LOSE
 G      PDX     ,2                      CAN PIECE BE MASKED
        LDI     KIND+1,2
        LFT     1
        TRA     F                       NO, PAWN KNIGHT OR KING
        PAI                             YES, ARE COLORS THE SAME
        ZET     KPIN+1,2                IS THIS PIECE PINNED
        TRA     PINK
 G1     IIS     INTER+1,1
        LFT     1
```

```
         TRA     F                NO
         ORA     =1               YES, TAG IT AND STORE
         STO     INTER,1          IT AFTER THE MASKER
         TXI     H,1,1            GO AROUND MASK LOOP AGAIN
*
 PIND    CLA     KPIN+1,2         PICK UP PIN INFO
         PDX     ,2               KING PIN IN DEC
         TMI     PIND1            INDICATES QUEEN PIN
 PIND2   PXD     ,4               GET DIRECTION
         ARS     6                NORMALIZE
         ADD     =1B17
         STO     PINDIR           SAVE DIRECTION
         PXD     ,2               REAL PIN DIRECTION
         CAS     PINDIR           ARE DIRECTIONS SAME
         TRA     *+2
         TRA     PIND3            DIRECTIONS MATCH
         CLA     IOPP+1,2
         CAS     PINDIR           ARE DIRECTIONS OPPOSITE
         TRA     *+2
         TRA     PIND3            OPPOSITE DIRECTION MATCHES
 PIND1   TXH     PIND4,2,0        DOUBLE PIN
         PAX     ,2               USE ADDRESS PART
         TRA     PIND2
 PIND3   PIA                      GET BACK PIECE
         PDX     ,2
         TRA     C1               PIECE OK TO USE
PIND4    LXA     F,2              THIS PIECE USELESS
         TXI     D,1,-1           GO BACK TO LOOP
PINDIR   PZE
*
 PINK    CLA     KPIN+1,2         PICK UP PIN INFO
         PDX     ,2               KING PIN IN DEC
         TMI     PINK1            INDICATES QUEEN PIN
 PINK2   PXD     ,4               GET DIRECTION
         ARS     6                NORMALIZE
         ADD     =1B17
         STO     PINDIR           SAVE DIRECTION
         PXD     ,2               REAL PIN DIRECTION
         CAS     PINDIR
         TRA     *+2
         TRA     PINK3            DIRECTIONS MATCH
         CLA     IOPP+1,2
         CAS     PINDIR
         TRA     *+2
         TRA     PINK3            OPPOSITE DIRECTION MATCHES
 PINK1   TXH     F,2,0            DOUBLE PIN
         PAX     ,2               USE ADDRESS PART
         TRA     PINK2
 PINK3   PIA                      GET BACK PIECE
         PDX     ,2
         TRA     G1               PIECE OK TO USE
PINK4    LXA     F,2              THIS PIECE USELESS
         TXI     D,1,-1           GO BACK TO LOOP
*        PHASE 2,  COPY INTER INTO IEXCH IN ORDER
```

```
ORDER   NZT     ATACK
        TRA     E                       NO ATTACKERS, FLUSH
        STZ     SIDE                    ATTACKERS-DEFENDERS FLIP-FLOP
        SXD     M,1                     END TEST FOR INTER TABLE
        LXD     K,2
        STO     IEXCH+1,2               SETS BEG OF PIECE
        STZ     COUNT1                  COUNTS TO NUM. ATT. OR DEF.
U       CLA     =2000B17
        STO     MINVAL                  +INFINITY = VALUE
        AXT     1,1
P       CLA     INTER+1,1               SEARCH FOR SMALLEST VALUED PIECE
        TMI     M-1                     PIECE USED
        LDI     INTER+1,1               SEPARATES ATTACKERS AND DEFENDERS
        IIS     SIDE                    ACCORDING TO SIDE
        IIS     K                       PIECES AGREEING WITH
        LFT     1                       SIDE GO TO Q
        TRA     Q
        TXI     *+1,1,1
M       TXL     P,1,**                  CLOSE INTER SEARCH LOOP
        CLA     MINVAL
        SUB     =2000B17
        TZE     NOMORE                  ALL ATTACKERS OR DEFENDERS USED
        LXD     COUNT,1                 INDEX TO IEXCH
CAND    AXT     **,4                    INDEX TO INTER
        CLA     INTER+1,4
        STO     IEXCH+1,1
        PDX     ,2              PICK UP THE BEARER FOR CHAINING
        LBT                             IS THIS PIECE MASKED
        TRA     SKIP                    NO
        CAL     INTER+2,4               YES, PICK UP COUNT1
        STP     IEXCH+1,1               MARK MASKED ENTRIES WITH MINUS SIGN
        STT     IEXCH+1,1               TAG IS INDEX OF MASKER (PREFIX -)
SKIP    CLS     COUNT1
        STO     INTER+1,4               STORE (-COUNT1)
        SUB     =1B20
        STT     COUNT1                  INCREMENT COUNT1
        ADD     =8B20
        TMI     LOSE                    TOO MANY ATT. OR DEF. ON ONE PIECE
        CLA     CHAIN+1,2               SETS THE CONNECTION OF DOUBLE
        PAX     ,4                      FUNCTION PIECES
        PXA     ,1
        STA     IEXCH+1,4
        STA     CHAIN+1,2
        TXI     *+1,1,1                 INCREMENTS COUNT
        SXD     COUNT,1
        TXL     U,1,128                 MAX SIZE OF IEXCH EXCEEDED
LOSE    CALL    ERROR,FMT
        TSX     $LDUMP,4
FMT     BCI     5,TABLE SIZE EXCEEDED BY SWAP.
        MTH     -1,7,-1
*       USED IN INTER LOOP
Q       PDX     ,2                      BEARER IN XR2
        CLA     KIND+1,2
        PDX     ,4
```

```
         CLA     KVAL+1,4          VALUE OF BEARER
         CAS     MINVAL
         TXI     M,1,1             TRA IF PIECE GREATER THAT MINVAL
         NOP
         LDI     INTER+1,1         IS PIECE MASKED
         RFT     1
         TRA     T                 YES
R1       STO     MINVAL            NO, STORE ITS VALUE AND
         SXA     CAND,1            ITS INTER INDEX.
         TXI     M,1,1             BACK TO INTER LOOP
T        LDI     INTER+2,1         HAS MASKER BEEN USED
         LNT     400000
         TXI     M,1,1             NO
         TRA     R1                YES
*        WE HAVE USED ALL ATTACKERS OR DEFENDERS.
NOMORE   LXD     K,1               ORIGINAL PIECE
         CLA     COUNT1            NUM ATT. OR DEF.
         ZET     SIDE
         TRA     V                 DEFENDERS
         STT     IEXCH+1,1         ATTACKERS
         CLA     =1B17
         STO     SIDE              FLIP SIDE
         TRA     U-1               PICK UP DEFENDERS
V        ALS     18
         STP     IEXCH+1,1
         TXI     W,1,1
*        STORAGE ALLOCATION
COUNT    PZE
COUNT1   PZE
SIDE     PZE
INDIC    PZE
ATACK    PZE
MINVAL   PZE
K        PZE
         COMMON  -206              SET TO TOP OF MEMORY
INTER    COMMON  20
CHAIN    COMMON  32
         COMMON  206-20-32+12561   SET TO 29000
R        COMMON  1
IBEAR    SYN     R+12307
LOC      SYN     R+10971
KIND     SYN     R+11099
KVAL     SYN     R+9645
IEXCH    SYN     R+3374
IOPP     SYN     R+11277
KPIN     SYN     R+6375
         END
```

```
*       LABEL
*       LIST8
        SUBROUTINE LTRADE(IW,IB,IND,IARG,IAT)
C       GIVEN A POSITION, AND UPDATED SWAP TABLES, COMPUTES THE MATERIAL
C       BALANCE VALUE OF THE POSITION AND SEVERAL STABILITY INDICATORS.
        DIMENSION MPVAL(32), N1AT(32)
        DIMENSION ITAB(16)
        DIMENSION FOO(5000)
        DIMENSION LOC(32),NFIRST(22),KPAWNV(8),IEXTD(16),IEXTS(64)
        DIMENSION IPIN(32),IOPP(16),KIND(32),MAVAIL(100),KVAL(6)
        DIMENSION IHOPE(64),IEXCH(128)
        DIMENSION LISP(6000)
        COMMON FOO
        EQUIVALENCE(FOO(2892),K1),(FOO(1463),KIND),(FOO(2765),MAVAIL)
        EQUIVALENCE(FOO(2900),MCOL)
        EQUIVALENCE(FOO(2703),IPIN),(FOO(1285),IOPP),(FOO(255),IBEAR)
        EQUIVALENCE(FOO(1365),IEXTD),(FOO(1301),IEXTS),(FOO(1527),IOCC)
        EQUIVALENCE(FOO(1591),LOC),(FOO(1623),NFIRST),(FOO(3003),KPAWNV)
        EQUIVALENCE (FOO(3121),PLY),(FOO(3120),IPE),(FOO(2917),KVAL)
        EQUIVALENCE (FOO(3051),MOBW),(FOO(3052),MOBB),(FOO(3123),IHOPE)
        EQUIVALENCE (FOO(9188),IEXCH),(FOO(3122),BACK),(FOO(3187),LISP)
        EQUIVALENCE (FOO(3053),MATW),(FOO(3054),MATB),(FOO(3119),MLOG)
        EQUIVALENCE (FOO(134),NLOG)
        MCOL=MCOL
        IARG = 0
        IAT=0
        IND=0
        IW=0
        IB=0
        IPLY=XSHIFTF(PLY,11)
        DO 5 I=1,32
        MPVAL(I)=0
    5   N1AT(I)=0
        DO 200 I=1,32
        NAT=XTAGF(IEXCH(I))
        IF(NAT)200,200,10
   10   NDEF=XPREF(IEXCH(I))
        IF(NAT-NDEF)20,20,30
   20   K = NAT+NAT+1
        GO TO 40
   30   K=NDEF+NDEF+2
   40   ITAB(1) = I
        IATOR=XDECF(IEXCH(I))
        IDEFOR=IATOR+NAT
        M=0
        J=1-XSHIFTF(XLBITF(I),1)
        IFAT = XDECF(IEXCH(IATOR))
        IDVAL=XGETF(KIND(I),KVAL)-XGETF(KIND(IFAT),KVAL)
        IF(IDVAL)50,50,57
   57   IF(XLBITF(KIND(IFAT)))50, 50, 400
  400   IAT=IAT+IDVAL*J
   50   DO 70 L=2,K,2
        ITAB(L)=XDECF(XGETF(M+IATOR,IEXCH))
        ITAB(L+1)=XDECF(XGETF(M+IDEFOR,IEXCH))
```

```
   70 M=M+1
      ITRA = 0
      DO 80 L=2,K
      JVALUE=XGETF(XGETF(ITAB(L-1),KIND),KVAL)
      ITAB(L-1)=ITRA
      ITRA=ITRA+XSIGNF(JVALUE,J)
   80 J=-J
      IF(J)100,100,90
  100 IF(K-2)130,130,105
  105 ITRA = XMAXOF(ITRA,ITAB(K-1))
      K=K-1
   90 IF(K-2)130,130,95
   95 ITRA=XMINOF(ITRA,ITAB(K-1))
      K=K-1
      GO TO 100
  130 IF(XLBITF(I))160,160,140
  140 ITRA = -ITRA
C     MPVAL(I) IS THE VALUE OF AN EXCHANGE SQUARE IF THE ATTACKER
C     INITIATES THE EXCHANGE WITH HIS LOWEST VALUED PIECE.  POSIVIVE
C     VALUES MEAN THE ATTACKER WINS MATERIAL.
  160 MPVAL(I) = ITRA
      IF(XLBITF(MCOL + I))163,163,161
  161 IF(ITRA)165,162,165
C     THE MOVER HAS AN EXCHANGE AVAILABLE TO HIM.
  162 IND = 1
      GO TO 165
  163 IF(ITRA)165,165,164
C     THE MOVER HAS A THREATENED PIECE.
  164 IARG=4-XMINOF(3,IPLY)
C     N1AT(I) IS THE NUMBER OF TIMES THAT PIECE I INITIATES AN EXCHANGE
C     SQUARE ATTACK.  IF IT IS GREATER THAN 1 WE HAVE A DOUBLE FUNCTION
C     PIECE.
  165 N1AT(IFAT) = N1AT(IFAT) + 1
  200 CONTINUE
      NCVAL = 0
      L1 = 3 - MCOL
      L2 = 30 + L1
      M2 = 30 + MCOL
      DO 300 I = L1, L2, 2
      IF(N1AT(I) - 1)300, 300, 240
  240 IF(IPLY - 3)250, 255, 255
  250 IF(XTAGF(IEXCH(I)))300,300,255
  255 DO 280 J9 = MCOL, M2, 2
      NAT = XTAGF(IEXCH(J9))
      IF(NAT)280,280,260
  260 IKE = XDECF(XGETF(XDECF(IEXCH(J9)),IEXCH))
      IF(IKE - I)280, 265, 280
  265 IF(IPLY - 3)270, 266, 266
  266 MPVAL(J9) = 0
      GO TO 310
  270 NCVAL = NCVAL + XMAXOF(0, MPVAL(J9))
  280 CONTINUE
  290 NCVAL=NCVAL + XMINOF(0, MPVAL(I))
      GO TO 310
```

```
  300 CONTINUE
  310 DO 320 I = MCOL, M2, 2
  320 NCVAL = XMAXOF(NCVAL, MPVAL(I))
      DO 330 I = 1, 31, 2
      IW = IW + XMAXOF(0, MPVAL(I))
  330 IB = IB + XMAXOF(0, MPVAL(I+1))
      IW = -IW
      GO TO (350, 380), MCOL
C     +IB OR -IW IS THE AMOUNT AN ATTACKER GAINS ON A BLACK OR WHITE
C     EXCHANGE SQUARE, TAKING INTO ACCOUNT THE VALUE OF THE SIDE TO MOVE
C     NCVAL IS THE BUGGER FACTOR WHICH ADJUSTS IB AND IW ACCORDING TO
C     THE SIDE TO MOVE.
C     NOTE THAT IB+IW IS THE EXPECTED MATERIAL VALUE OF THE POSITION.
  350 IW = IW + NCVAL
      GO TO 230
  380 IB = IB - NCVAL
  230 IAT=XSIGNF(XONEF(IAT),IAT)
      RETURN
      END
```

```
*       LABEL
*       LIST8
        SUBROUTINE PINS
        COMMON FOO
        DIMENSION IBEAR(64,16), IOCC(64)
        DIMENSION FOO(5000)
        DIMENSION LOC(32),NFIRST(22),KPAWNV(8),IEXTD(16),IEXTS(64)
        DIMENSION IPIN(32),IOPP(16),KIND(32),MAVAIL(100),KVAL(6)
        DIMENSION  KPIN(32)
        EQUIVALENCE(FOO(2892),K1),(FOO(1463),KIND),(FOO(2765),MAVAIL)
        EQUIVALENCE(FOO(2900),MCOL)
        EQUIVALENCE(FOO(2703),IPIN),(FOO(1285),IOPP),(FOO(255),IBEAR)
        EQUIVALENCE(FOO(1365),IEXTD),(FOO(1301),IEXTS),(FOO(1527),IOCC)
        EQUIVALENCE(FOO(1591),LOC),(FOO(1623),NFIRST),(FOO(3003),KPAWNV)
        EQUIVALENCE (FOO(3121),PLY),(FOO(3120),IPE),(FOO(2917),KVAL)
        EQUIVALENCE (FOO(6187), KPIN)
        DO 40  J = 1,32
40      KPIN(J) = 0
        DO 20  J = 1,2
        KRAP = 1
        GO TO 7
20      CONTINUE
        DO 30  J = 31,32
        KRAP = 2
        GO TO 7
30      CONTINUE
        RETURN
7       KLOC = LOC(J)
        DO 1  I = 1,8
        JPIN = LOOK (KLOC, IOPP(I))
        IF (JPIN) 1,1,3
3       IF (XLBITF (IOCC(JPIN)+J))  1,4,1
4       IFOO = IBEAR(JPIN,I)
        IF (IFOO) 1,1,5
5       IF (XORF(XLBITF(IFOO+J), XLBITF(KIND(IFOO)+1))) 6,1,6
6       JPIN = IOCC(JPIN)
        GO TO (15, 16), KRAP
15      KPIN(JPIN) = I
        GO TO 1
16      KPIN(JPIN) = -(KPIN(JPIN) + XSHIFTF(1,-18))
1       CONTINUE
        GO TO (20, 30), KRAP
        END
```

```
*       LABEL
*       FAP
*PAWN    STRUCTURE FOR CHESS, MARCH 2, 1962
        COUNT   150
        ENTRY   JPAWNS
JPAWNS  SXA     XR1,1
        SXA     XR2,2
        SXD     XR4,4
XR4     SYN     JPAWNS-2
*       EMTM                            FOR 7094
        CLA     NOP                     SET UP FOR WHITE LOOP
        STO     COLOR
        STA     COLOR1
        AXT     NP3+1,4
        SXA     NP3,4
        CLA     TABLE                   INITIALIZE XECUTE FOR WHITE
        LDQ     TABLE-8
        AXT     TABLE,1
        AXT     COLOR2+1,4
LOOP    SXA     COLOR2,4                MAJOR LOOP, EXEC. FOR BLACK AND WHITE
        SXA     SET1,1
        SXA     SET2,1
        STD     TABLE-6
        SLQ     TABLE-22
        STZ     PAWNV
        AXT     0,1                     FILE IN XR1
FILEL   STZ     ADJAC                   ADJACENT PAWN INDICATOR
        STZ     PROTEC                  ANOTHER PAWN PROTECTING INDIC.
        STZ     NPAWNS
        STZ     PAST                    INDICATED A PASSED PAWN
        CLS     =2B17
        STO     OTHER
COLOR1  AXT     **,2                    RANK IN XR2, 0 FOR WH., 56 FOR BLK.
RANKL   TXL     NP1,1,0
        CLA     IOCC+1,3
        PDX     ,4
SET1    XEC     **,4
NP1     TXH     NP2,1,6
        CLA     IOCC-1,3                ONLY IF A FILE EXISTS TO LEFT
        PDX     ,4
SET2    XEC     **,4
NP2     CLA     IOCC,3
        PDX     ,4
        TXL     NP3,4,6
        TXH     NP3,4,22
        PXA     ,4
COLOR   HTR     *
        LBT
        TRA     OPPOS                   THIS IS AN OPPOSITION PAWN
        CLA     NPAWNS
        ADD     =1B17
        STO     NPAWNS
        CLA     ADJAC
        STO     PROTEC
```

```
        SXA     LRANK,2         THIS SAVES THE ABSOLUTE RANK
        STZ     PAST
        TRA     NP3
OPPOS   STL     PAST
        STZ     OTHER
NP3     TXI     **,2,8          *+1 FOR WHITE, BLACK FOR BLACK
        TXL     RANKL,2,56
EVL     CLA     PAWNV           EVALUATOR
        LXD     NPAWNS,2
        TXH     *+3,2,0
        ADD     IOPEN,1         OPEN FILE
        TRA     CONT1
        NZT     PROTEC
        ADD     OTHER
        NZT     PROTEC
        ADD     IBKWD,1
        NZT     ADJAC
        ADD     ISOLAT,1
        TXL     NDBL,2,1        NOT DOUBLED PAWN
        ADD     IDBLD,1         DOUBLED PAWN
        LXA     LRANK,2
        TXL     *+4,2,23
        SUB     =1B17
        TXL     *+2,2,24
        SUB     =1B17
        ADD     OTHER
NDBL    LXA     PAST,2
        TXH     CONT1,2,0
        ADD     =10B17
        LXA     LRANK,2
        TXL     *+2,2,24
        ADD     =1B17
        ADD     KPAST,1         PAST PAWN
CONT1   STQ     PAWNV           THIS HAS BEEN IN AC ALL THIS TIME
        TXI     *+1,1,1
        TXL     FILEL,1,7
COLOR2  TRA     *               *+1 FOR WHITE, DONE FOR BLACK
        STO     TPAWNV
        CLA     COM
        STO     COLOR
        AXT     56,4
        SXA     COLOR1,4
        AXT     BLACK,4         RE-INITIALIZE FOR BLACK
        SXA     NP3,4
        AXT     DONE,4
        AXT     TABLE+1,1
        CLA     TABLE-8
        LDQ     TABLE
        TRA     LOOP
COM     COM
NOP     NOP     0
BLACK   TXI     *+1,2,-16
        TXL     RANKL,2,-9
        LAC     LRANK,2
```

```
        TXI     *+1,2,56            CONVERT INTO TRUE RANK
        SXA     LRANK,2
        TRA     EVL
DONE    CLA     TPAWNV
        SUB     PAWNV
*       LMTM                        FOR 7094
XR1     AXT     **,1
XR2     AXT     **,2
        LXD     XR4,4
        TRA     2,4
        DUP     1,10
        PDX     ,0
        STL     PAST
        DUP     2,8
        STL     ADJAC
        STL     PAST
        DUP     1,7
        PDX     ,0
TABLE   SYN     *-2
PAWNV   PZE
ADJAC   PZE
PROTEC  PZE
NPAWNS  PZE
PAST    PZE
OTHER   PZE
TPAWNV  PZE
LRANK   PZE
*       VALUE TABLES
        DEC     7B17,7B17,8B17,8B17,8B17,8B17,7B17
IOPEN   DEC     7B17
        DEC     -5B17,-1B17,-1B17,-1B17,-1B17,-1B17,-1B17
ISOLAT  DEC     -5B17
        DEC     0,-5B17,-5B17,-6B17,-6B17,-5B17,-5B17
IBKWD   DEC     0
        DEC     -4B17,-4B17,-2B17,-3B17,-3B17,-2B17,-4B17
IDBLD   DEC     -4B17
        DEC     -3B17,0,0,0,0,0,0
KPAST   DEC     -3B17
ZILCH   COMMON  12561
R       COMMON  1
IOCC    SYN     R+11035
        END
```

```
*       LABEL
*        LIST8
        FUNCTION ICENTR(I123)
C       COMPUTES THE CENTER CONTROL FUNCTION.  LCENSQ IS A TABLE OF CENTER
C       SQUARES.  KCNVAL IS A TABLE OF RELATIVE WEIGHTS OF THOSE SQUARES.
        COMMON FOO
        DIMENSION  KPIN(32)
        DIMENSION IBEAR(64,16), LOC(32), KIND(32), FOO(5000)
        DIMENSION LCENSQ(16), KCNVAL(16)
        EQUIVALENCE (FOO(9317), NMOVES)
        EQUIVALENCE (FOO(6187), KPIN)
        EQUIVALENCE(FOO(2892),K1),(FOO(1463),KIND),(FOO(2765),MAVAIL)
        EQUIVALENCE(FOO(2703),IPIN),(FOO(1285),IOPP),(FOO(255),IBEAR)
        EQUIVALENCE(FOO(1591),LOC),(FOO(1623),NFIRST),(FOO(3003),KPAWNV)
        EQUIVALENCE (FOO(3011), LCENSQ), (FOO(3027), KCNVAL)
        ICENTR = 0
        IF (NMOVES - 30) 102, 101, 101
102     I123 = I123
        DO 100 I = 1,16
        K = LCENSQ(I)
        DO 90 J = 1, 16
        IF(IBEAR(K,J))90, 90, 10
   10   KP = IBEAR(K,J)
        IF (KPIN(KP)) 90,13,90
   13   IF(KIND(KP) - 6)15,110,15
  110   IF(XLBITF(KP))130,130,120
  120   ICENTR = ICENTR + KCNVAL(I)/3
        GO TO 40
  130   ICENTR = ICENTR - XGETF(17-I, KCNVAL)/3
        GO TO 40
   15   IF(XLBITF(KP))30, 30, 20
   20   ICENTR = ICENTR + KCNVAL(I)
        GO TO 40
   30   ICENTR = ICENTR - XGETF(17-I, KCNVAL)
   40   LOCKP = LOC(KP)
        IF(IBEAR(LOCKP, J))90, 90, 50
   50   KPP = IBEAR(LOCKP, J)
        IF(XLBITF(KPP+KP) + XLBITF(KIND(KPP)))90, 60, 90
   60   KP = KPP
        GO TO 15
   90   CONTINUE
  100   CONTINUE
        ICENTR = (ICENTR * XMINOF (10, 30 - NMOVES))/10
101     RETURN
        END
```

```
*       LABEL
*       LIST8
        FUNCTION IDVLOP(I123)
C       COMPUTES THE STATIC EVALUATION FUNCTION FOR DEVELOPMENT
        DIMENSION FOO(6000), LOC(32), NFIRST(22), KPAWNV(8), IEXTD(16)
        DIMENSION IEXTS(64), IOCC(64)
        COMMON FOO
        EQUIVALENCE (FOO(9317), NMOVES)
        EQUIVALENCE(FOO(2892),K1),(FOO(1463),KIND),(FOO(2765),MAVAIL)
        EQUIVALENCE(FOO(2900),MCOL)
        EQUIVALENCE(FOO(2703),IPIN),(FOO(1285),IOPP),(FOO(255),IBEAR)
        EQUIVALENCE(FOO(1365),IEXTD),(FOO(1301),IEXTS),(FOO(1527),IOCC)
        EQUIVALENCE(FOO(1591),LOC),(FOO(1623),NFIRST),(FOO(3003),KPAWNV)
        XBLTCHF(J)=XORF(XGETF(J+ICOLOR,LOC),XTRANKF(XGETF(J+ICOLOR,LOC),
       1J+ICOLOR)) + XNOTF(XGETF(J+ICOLOR,LOC))
        IDVLOP = 0
        I123 = I123
        ICOLOR = 0
        IF (NMOVES - 15) 69, 100, 100
   69   IBARF = IPESS
        IPESS = 0
        DO 1 I = 7, 21, 2
    1   IPESS = IPESS+XNOTF(XGETF(I+ICOLOR,NFIRST))
        DO 2 I = 13,15,2
        IPESS = IPESS+XGETF(XBLTCHF(I),KPAWNV)
        IF(XGETF(I+ICOLOR,NFIRST)+XNOTF(XGETF(I+ICOLOR,LOC)))2,22,2
   22   IDIR = XSHIFTF(ICOLOR+1,1)
        NSQ = XMOVF(IEXTD(IDIR)+XGETF(XGETF(ICOLOR+I,LOC),IEXTS))
        IF(IOCC(NSQ)+XGETF(XMOVF(IEXTD(IDIR)+IEXTS(NSQ)),IOCC))23,2,23
   23   IPESS = IPESS - 5
    2   CONTINUE
        IPESS = IPESS + 5*XNOTF(XBLTCHF(11)-4)
        IF (ICOLOR)40,40,50
   40   KJ1 = 2
        KJ2=7
        KQ2 = 12
        GO TO 60
   50   KJ1=58
        KJ2=63
        KQ2 = 52
   60   IF(IOCC(KJ1)-23-ICOLOR)62,61,62
   62   IPESS = IPESS + 4
   61   IF(IOCC(KJ2)-25-ICOLOR)64,63,64
   64   IPESS = IPESS + 4
   63   IF(IOCC(KJ1+1)-27-ICOLOR)66,65,66
   66   IPESS = IPESS + 3
        IF (IOCC(KQ2) -23 -ICOLOR) 65,166,65
  166   IPESS = IPESS -10
   65   IF(IOCC(KJ2-1)-29-ICOLOR)68,67,68
   68   IPESS = IPESS + 3
        IF(IOCC(KQ2+1) -25 -ICOLOR) 67,168,67
  168   IPESS = IPESS -10
   67   IF(IOCC(KJ1+2)-31-ICOLOR)71,70,71
   70   IPESS = IPESS + 7
```

```
      GO TO 75
   71 IPESS=4*XORF(LOC(ICOLOR+31),XNOTF(XRANGEF(XBLTCHF(31),1,3)))+IPESS
   75 ICOLOR = ICOLOR + 1
      GO TO (69,711) ,ICOLOR
  711 IDVLOP = IBARF - IPESS
100   RETURN
      END
```

61

```
*       LABEL
*       LIST8
        SUBROUTINE REPLY5
        DIMENSION MPVAL(100)
        DIMENSION FOO(5000)
        DIMENSION LOC(32),NFIRST(22),KPAWNV(8),IEXTD(16),IEXTS(64)
        DIMENSION IPIN(32),IOPP(16),KIND(32),MAVAIL(100),KVAL(6)
        DIMENSION IHOPE(64),IEXCH(128)
        DIMENSION LISP(6000)
        DIMENSION KPLY(20)
        COMMON FOO
        EQUIVALENCE (FOO(6219),IWHTM),(FOO(6220),IBLKM)
        EQUIVALENCE(FOO(2892),K1),(FOO(1463),KIND),(FOO(2765),MAVAIL)
        EQUIVALENCE(FOO(2900),MCOL)
        EQUIVALENCE(FOO(2703),IPIN),(FOO(1285),IOPP),(FOO(255),IBEAR)
        EQUIVALENCE(FOO(1365),IEXTD),(FOO(1301),IEXTS),(FOO(1527),IOCC)
        EQUIVALENCE(FOO(1591),LOC),(FOO(1623),NFIRST),(FOO(3003),KPAWNV)
        EQUIVALENCE (FOO(3121),PLY),(FOO(3120),IPE),(FOO(2917),KVAL)
        EQUIVALENCE (FOO(3051),MOBW),(FOO(3052),MOBB),(FOO(3123),IHOPE)
        EQUIVALENCE (FOO(9188),IEXCH),(FOO(3122),BACK),(FOO(3187),LISP)
        EQUIVALENCE (FOO(3053),MATW),(FOO(3054),MATB),(FOO(3119),MLOG)
        EQUIVALENCE (FOO(134),NLOG)
        EQUIVALENCE(FOO(2877),ICHECK)
        EQUIVALENCE (KPLY,FOO(9167))
   10   IF(K1) 31,31,20
   20   J=-MCOL-MCOL+3
        IPLY=XSHIFTF(PLY,11)
        ISTAB = 0
        ID = IDVLOP(1)
 21     IPE = XMINOF(KPLY(IPLY),K1)
        IF (IPE) 666, 666, 99
   99   IF(IPLY-2)30,30,200
   30   ISTAB=1
  600   DO 80 M=1,K1
        NP=XGETF(XMV1F(MAVAIL(M)),IOCC)
        MVR=XMV3F(MAVAIL(M))
        IF(KIND(MVR)-5)35,32,35
   32   IF(XABSF(LOC(MVR)-XMV1F(MAVAIL(M)))-2)35,33,35
   33   KS=28
        GO TO 37
   35   KS=0
   37   CALL UPDATE(MAVAIL(M))
  869   CALL PINS
        CALL SWAP
        CALL LTRADE(IW,IB,IND,IARG,IAT)
        IDT = IDVLOP(1)
        IF(IAT*J)62,62,60
   60   IF(XMAXOF((IDT-ID)*J-2,0)+XNOTF(XRANGEF(MVR,13,16)))62,62,61
   61   ISTAB = 1
        GO TO 629
   62   IAT=0
  629   IF (NP)50,50,40
   40   MVAL=XGETF(KIND(NP),KVAL)
        IKAPT = 6
```

```
      GO TO 70
   50 MVAL = 0
      IKAPT = 0
70    IF(J) 555, 556, 556
555   NVAL = MVAL * IWHTM
      GO TO 77
556   NVAL = MVAL * IBLKM
77    MPVAL = NVAL + (IWHTM * IW + IBLKM * IB +XSHIFTF (IDT, 2) + ICENTR
     1(1) + XSHIFTF (IAT, 4) + 3*JPAWNS (1))*J + KS + 24/K1**2 + IKAPT +
     2IARG
   80 CALL REVERT
      IF (ISTAB) 250,250,85
  250 IF(XLBITF(IPLY))85,85,31
   85 DO 120 I=1,IPE
      LM=IPE-I+1
      MVAL=-5000
      DO 110 M=1,K1
      IF(MPVAL(M)-MVAL)110,110,90
   90 MVAL=MPVAL(M)
      K=M
  110 CONTINUE
      IHOPE(LM)=MAVAIL(K)
  120 MPVAL(K)=-5000
      GO TO 900
  200 CALL SWAP
      CALL LTRADE(IW, IB, IND, IARG, IAT)
      IF(IND+IB-IW+IARG)600,600,210
  210 IF(IPLY-3)30,30,220
  220 IF(IB-IW+XABSF(IARG))600,600,222
  222 IF(IPLY-5)30,30,224
  224 IF(IB-IW)600,600,30
  900 IF(IPLY-2)905,905,950
 1000 FORMAT(6H0IPLY=I6,4X,14A6)
  905 DO 910 M=1,IPE
  910 CALL JUNPAK(IHOPE(M),MPVAL(M),MPVAL(M+8))
      WRITE OUTPUT TAPE 100,1000,IPLY,((MPVAL(M),MPVAL(M+8)),M=1,IPE)
      GO TO 950
  666 WRITE OUTPUT TAPE 100,1000,IPLY
   31 IPE=0
  950 RETURN
      END
```

```
*       LABEL
*       LIST8
        SUBROUTINE EVAL
        DIMENSION FOO(5000)
        DIMENSION LOC(32),NFIRST(22),KPAWNV(8),IEXTD(16),IEXTS(64)
        DIMENSION IPIN(32),IOPP(16),KIND(32),MAVAIL(100),KVAL(6)
        DIMENSION IHOPE(64),IEXCH(128)
        DIMENSION LISP(6000)
        DIMENSION NTYPE(50)
        COMMON FOO
        EQUIVALENCE(FOO(2892),K1),(FOO(1463),KIND),(FOO(2765),MAVAIL)
        EQUIVALENCE (FOO(6219),IWHTM),(FOO(6220),IBLKM)
        EQUIVALENCE(FOO(2900),MCOL)
        EQUIVALENCE(FOO(2703),IPIN),(FOO(1285),IOPP),(FOO(255),IBEAR)
        EQUIVALENCE(FOO(1365),IEXTD),(FOO(1301),IEXTS),(FOO(1527),IOCC)
        EQUIVALENCE(FOO(1591),LOC),(FOO(1623),NFIRST),(FOO(3003),KPAWNV)
        EQUIVALENCE (FOO(3121),PLY),(FOO(3120),IPE),(FOO(2917),KVAL)
        EQUIVALENCE (FOO(3051),MOBW),(FOO(3052),MOBB),(FOO(3123),IHOPE)
        EQUIVALENCE (FOO(9188),IEXCH),(FOO(3122),BACK),(FOO(3187),LISP)
        EQUIVALENCE (FOO(3053),MATW),(FOO(3054),MATB),(FOO(3119),MLOG)
        EQUIVALENCE (FOO(134),NLOG)
        EQUIVALENCE (I,A)
        EQUIVALENCE (FOO(2913),NSPEC),(FOO(2649),NTYPE)
        IF(K1)10,10,15
   10   I=XSIGNF(10000,MCOL+MCOL-3)
        GO TO 30
   15   KS=0
        CALL PINS
   60   CALL SWAP
        CALL LTRADE(IW,IB,IND,IARG,IAT)
   50   IF(NSPEC)20,20,7
    7   DO 1 I=1,NSPEC
        IF(NTYPE(I)+1)8,1,1
    8   IF(XLBITF(XMV3F(NTYPE(I))))4,4,5
    4   KS=KS-28
        GO TO 1
    5   KS=KS+28
    1   CONTINUE
20      I = IWHTM*(MATW+IW)+IBLKM*(IB-MATB)+3*JPAWNS(1)+XSHIFTF(IDVLOP(1),
       12)+ICENTR(1)
B   30  A=A
        RETURN
        END
```

```
*       LABEL
*       LIST8
C       THE LONG AWAITED STRATEGY PROGRAM.  MAY 1, 1962
        SUBROUTINE STRTGY
        COMMON FOO
        EQUIVALENCE (FOO(3053),MATW), (FOO(3054),MATB)
        EQUIVALENCE (FOO(6219),IWHTM),(FOO(6220),IBLKM)
        CALL PINS
        CALL SWAP
        CALL LTRADE (IW,IB,IND,IARG,IAT)
        ITEM = IW + IB + MATW - MATB
        IWHTM = 60
        IBLKM = 60
        IF (XABSF (ITEM) - 4) 1, 2, 2
2       IWHTM = IWHTM -XSIGNF (10, ITEM)
        IBLKM = IBLKM +XSIGNF (10, ITEM)
1       RETURN
        END
```

```
*       LABEL
*       FAP
        COUNT   31
*       ALIAS,  UPDATE,REVERT,CCOL,SETUP
        ENTRY   UPDATE
        ENTRY   REVERT
        ENTRY   CCOL
        ENTRY   SETUP
UPDATE  SXD     UPDATE-2,4
        CLA*    1,4
        TZE     ZERO
        STO     MIN
        CALL    UPREV,MIN,ONE
RTN     LXD     UPDATE-2,4
        TRA     2,4
ZERO    CALL    ERROR,FMT
        TRA     RTN
FMT     BCI     5, UPDATE CALLED WITH ZERO ARG.
        MTH     -1,7,-1
REVERT  SXD     UPDATE-2,4
        CALL    UPREV,ZRO,TWO
RTN1    LXD     UPDATE-2,4
        TRA     1,4
CCOL    SXD     UPDATE-2,4
        CALL    UPREV,ZRO,FOR
        TRA     RTN1
SETUP   SXD     UPDATE-2,4
        CALL    UPREV,MIN3,THR
        TRA     RTN1
ZRO     PZE
ONE     PZE     ,,1
TWO     PZE     ,,2
THR     PZE     ,,3
FOR     PZE     ,,4
MIN3    MZE     ,,3
MIN     PZE
        END
```

```
*       LABEL
*       LIST8
C       UPREV CHESS SUBROUTINE, 2/26/62, MINOR REVISION
        SUBROUTINE UPREV(MIN,M6)
C       DIMENSION AND EQUIVALENCE STATEMENTS
        DIMENSION IOCC(64),LOC(32),NFIRST(22),NUMB(50),
       1NTYPE(50),IBEG(33),IEND(32),MOVE(304),ICAPT(150),
       2MOVEFR(150),MOVEP(150),JBEAR(1024),IBEAR(64,16),
       3KIND(32),MSVN(16),IPDIR(3,2),IEXTD(16),IEXTS(64),
       4M64M1(16),NMOV(6),IOPP(16)
        DIMENSION JPAWN(8)
        DIMENSION MSTO(32)
        DIMENSION MAVAIL(100),ITCH(2),ITCHD(2),IPIN(32)
        DIMENSION NEP(10),MEP1(10),MEP2(10)
        DIMENSION JPROM(4)
        DIMENSION LOGG(101)
        DIMENSION NZZZ(120)
        DIMENSION KVAL(6),KFORCE(64),KWORTH(64)
C               COMMON STATEMENTS
        COMMON IPDIR,IOPP,IEXTS,IEXTD,JPAWN,M64M1,MSVN,NMOV,MSTO,JPROM,
       1IBEAR,JBEAR,KIND,IEND,IBEG,IOCC,LOC,NFIRST,MOVE,IENDS,MOVEP,
       2MOVEFR,ICAPT,NUMB,NTYPE,ITCH,ITCHD,IPIN,NEP,MEP1,MEP2,LOGG,NLOG,
       3NZZZ,NUMTES,MAVAIL,IZ,IY,IX,IU,IT,ISPEC,IR,IQ,IPROM,IOPPD,INTER,
       4IDIR,ICHECK,IA,IAA,A,JA,JB,JC,JDIR,JD,JE,JF,JIN,JJ,JROOK,J,K1,KD,
       5K,L2,L,M4,MAREA,MCAPT,MCOL,MIN,MOVDIR,MOVENO,MOVER,MOVETO,MQ,M,
       6MVR,N1,N2,NEWSQ,N,NSPEC,NUMEP,NPRINT,KIN,KVAL,KFORCE,KWORTH,MOBW,
       7MOBB,MATW,MATB
        EQUIVALENCE (IENDS,IBEG(33)),(NLOG,LOGG(101)),(NUMTES,NZZZ(120)),
       1(IBEAR,JBEAR)
        DIMENSION NUMBER(64)
        COMMON NUMBER
        COMMON MLOG
C
C               MAIN PROGRAM
C
        MCOL = MCOL
        GO TO (120,150,700,200),M6
C       CHANGE COLOR OF SIDE TO MOVE
 200    MCOL=3-MCOL
        MIN=-MCOL
        MOVENO=MOVENO+1
        MOVEP(MOVENO)=-1
        GO TO 700
C       MIN IS THE MOVE MADE
 120    MOVETO = XMV1F(MIN)
        MOVDIR = XMV2F(MIN)
        MOVER = XMV3F(MIN)
C
C
C       SET UP VARIABLES FOR UPDATE
C
 130    MQ=LOC(MOVER)
        KD=KIND(MOVER)
        MOVENO=MOVENO+1
```

```
      MCOL = 1 + XLBITF(MOVER)
C
C     BRANCH ON PIECE KIND
      GO TO (400,131,134,134,460,134),KD
C          THIS MAY BE THE FIRST MOVE OF A ROOK
 131  IF(NFIRST(MOVER))133,133,134
 133  NFIRST(MOVER)=1
      NSPEC=NSPEC+1
      NUMB(NSPEC)=MOVENO
      NTYPE(NSPEC)=1
C     IS THE MOVE A CAPTURE
 134  IF(IOCC(MOVETO))137,137,136
C     CAPTURE
 136  ICAPT(MOVENO)=IOCC(MOVETO)
      CALL PUTCH(IOCC(MOVETO),0)
137   CALL PUTCH(MOVER,MOVETO)
 139  MOVEFR(MOVENO)=MQ
 141  MOVEP(MOVENO)=MOVER
C
C
C
C          1. CHECKS AND PINS
C          2. LIST LEGAL MOVES OF KINGS DIRECTLY IN MAVAIL TABLE
C          3. LIST MOVES OF THE OTHER PIECES IN THE MAVAIL TABLE
C
C               INITIALIZE.
700   ICHECK = 0
      KLOC = LOC(MCOL)
      DO 701 JA=1,32
701   IPIN(JA) = 0
      DO 702 I=1,2
      ITCH(I)=0
 702  ITCHD(I)=0
      IR = IEXTS(KLOC)
      K1 = 0
      M = IBEG(MCOL) - 1
C              END OF INITIALIZATION
C
C              IS THE KING IN CHECK.  LIST PINS.
      DO 715 K=1,16
C     IS THE KING SUBJECT TO CAPTURE BY THE OTHER SIDE
      IF (IBEAR(KLOC,K)) 721,721,718
C              HAS THE BEARER THE SAME COLOR AS THE KING.
 718  IF (XLBITF(IBEAR(KLOC,K)+MCOL)) 750,716,750
C
C          THE KING IS IN CHECK.
750   ICHECK = ICHECK + 1
      ITCH(ICHECK) = IBEAR(KLOC,K)
      ITCHD(ICHECK) = IOPP(K)
      IF(ICHECK - 2)715,731,731
C
C          KNIGHTS CANNOT PIN
721   IF(K - 8)722,722,715
722   IQ = XGETF(IOPP(K),IEXTD)
```

```
      IZ = IR
C
C           LOOK FOR OCCUPIED SQUARE ALONG LINE FROM KING
728   IZ = IZ +IQ
      NEWSQ=XMOVF(IZ)
      IF (NEWSQ-64) 719,719,715
 719  IF (IOCC(NEWSQ)) 728,728,727
C
C       AN OCCUPIED SQUARE IS FOUND
716   NEWSQ = XGETF(IBEAR(KLOC,K),LOC)
C     FIND WHAT IF ANYTHING BEARS FROM OPPOSITE DIRECTION
727   IU = IBEAR(NEWSQ,K)
      IF (IU) 715,715,726
C
C           IF BEARER IS A LONG RANGE PIECE OF OPPOSITE COLOR WE GET A
C     PIN.
726   IF(1-XLBITF(IU+MCOL)+XLBITF(KIND(IU)))715,732,715
C
C     LIST A PIN
 732  IT=IOCC(NEWSQ)
      IPIN(IT) = K
 715  CONTINUE
C
C
C
C     PUT MOVES OF KINGS IN MAVAIL TABLE
C     FIRST NON-CASTLING MOVES
 731  DO 705 IDIR=1,8
      IF(XGETF(M+IDIR,MOVE))705,705,706
706   NEWSQ = XMV1F(XGETF(M+IDIR,MOVE))
C           THE KING CANNOT MOVE ALONG THE LINE OF CHECK,
C           UNLESS THE CHECKER IS A PAWN.
      IF (ICHECK) 753,708,753
 753  DO 751 JA=1,ICHECK
      IF (ITCHD(JA)-IOPP(IDIR)) 751,752,751
 752  IF (XGETF(ITCH(JA),KIND)-1) 705,751,705
 751  CONTINUE
 708  DO 712 K=1,16
      IF(IBEAR (NEWSQ,K))712,712,713
 713  IF (XLBITF(IBEAR(NEWSQ,K)+MCOL)) 705,712,705
 712  CONTINUE
      K1=K1+1
      MAVAIL(K1)=XGETF(M+IDIR,MOVE)
 705  CONTINUE
C     ARE THERE CASTLING MOVES
C     NOT IF KING IS IN CHECK OR HAS MOVED
      IF(ICHECK+NFIRST(MCOL))800,736,800
C
C     FOR EACH ROOK
 736  DO 737 IDIR=1,3,2
C
C     DOES A ROOK WHICH HAS NEVER MOVED BEAR ON THE KING
      IF(IBEAR(KLOC,IDIR))739,737,739
 739  JROOK=IBEAR(KLOC,IDIR)
```

```
      IF(KIND(JROOK)-2+NFIRST(JROOK))737,738,737
C
C     ARE THE INTERMEDIATE SQUARES COVERED BY THE FOE
 738  JB=IDIR-2
      JD=KLOC
C
C     FOR EACH SQUARE THE KING MOVES OVER
      DO 741 JC=1,2
      JD=JD+JB
C
C     FOR EACH DIRECTION FROM THE INTERMEDIATE SQUARE
      DO 742 JDIR=2,16
      JE=IBEAR(JD,JDIR)
      IF (JE) 742,742,744
744   IF(XLBITF(MCOL + JE))737,742,737
 742  CONTINUE
 741  CONTINUE
C
C     CASTLING OK
      K1=K1+1
      MAVAIL(K1)=JD+XGETF(IOPP(IDIR),M64M1)+MSTO(MCOL)
 737  CONTINUE
C
C
C     MOVES OF OTHER PIECES IN MAVAIL TABLE, OMITTING KINGS
 800  K=MCOL+2
      IF(ICHECK-1)802,824,825
802   DO 803 I=K,32,2
      IF(LOC(I))804,803,804
804   M= IBEG(I)
C           IF A PAWN HAS MOVED, IT CANNOT ADVANCE TWO SQUARES.
      IF (XMAXOF(KIND(I)-1,1-NFIRST(I))) 815,816,815
 816  N=IEND(I)-1
      GO TO 817
 815  N=IEND(I)
C        IS PIECE PINNED
 817  IF (IPIN(I)) 805,806,805
C        NO PIN
806   DO 807 J=M,N
      IF(MOVE(J))807,807,808
808   K1 = K1+1
      MAVAIL(K1) = MOVE(J)
807   CONTINUE
      GO TO 803
C           PINNED
805   IDIR = IPIN(I)
      IOPPD = IOPP(IDIR)
809   DO 812 J=M,N
      IF(MOVE(J))812,812,813
813   IF(XMINOF(XABSF(XMV2F(MOVE(J))-IDIR),XABSF(XMV2F(MOVE(J))-IOPPD)))
     1812,814,812
814   K1=K1+1
      MAVAIL(K1) = MOVE(J)
812   CONTINUE
```

```
803   CONTINUE
C               ADJOIN EN PASSANT MOVES IF ANY
      IF(NUMEP)860,143,860
860   IF(NEP(NUMEP)-MOVENO)143,850,143
 850  JJ=1
859   GO TO (851,852,143),JJ
851   J1 = MEP1(NUMEP)
      GO TO 853
852   J1 = MEP2(NUMEP)
      IF(J1)853,143,853
C               IS THE EN PASSANT MOVE PREVENTED BY A PIN
853   IF(XGETF(XMV3F(J1),IPIN))854,855,854
C               PINNED, WHAT ABOUT THE DIRECTION.
854   IF(XMINOF(XABSF(XGETF(XMV3F(J1),IPIN)-XMV2F(J1)),XABSF(XGETF(XGETF
     1(XMV3F(J1),IPIN),IOPP)-XMV2F(J1))))856,855,856
C               NO PIN ON MOVE.  WILL REMOVAL OF CAPTURED PAWN PUT US
C     IN CHECK.
 855  IF (XRANKF(KLOC)-XRANKF(XGETF(XMV3F(J1),LOC))) 858,857,858
C     KING ON SAME RANK AS PAWNS.  REFERENCES TO PUTCH ARE NEEDED
C     TO REMOVE PAWNS FROM POSSIBLE LINE OF ACTION.
 857  J1OCC=XMV3F(J1)
      J1LOC=LOC(J1OCC)
      J2=XMV2F(J1)
      J3=XMOVF(IEXTS(J1LOC)+XGETF(4-XABSF(13-J2-J2),IEXTD))
      J3OCC=IOCC(J3)
      CALL PUTCH(J1OCC,0)
      CALL PUTCH(J3OCC,0)
      DO 864 K=1,3,2
      IF (IBEAR(KLOC,K)) 864,864,861
 861  IF (XLBITF(IBEAR(KLOC,K)+MCOL)) 864,864,862
 864  CONTINUE
      J4=0
      GO TO 863
 862  J4=1
 863  CALL PUTCH(J1OCC,J1LOC)
      CALL PUTCH(J3OCC,J3)
      IF (J4) 858,858,856
C               PUT EN PASSANT MOVE IN MAVAIL.
858   K1 = K1 + 1
      MAVAIL(K1) = J1
 856  JJ=JJ+1
      GO TO 859
C
C     SINGLE CHECK LEGAL KING MOVES HAVE
C     ALREADY BEEN FOUND.  LOOK FOR INTERPOSITIONS OR
C     CAPTURE OF CHECKER ALONG CHECK LINE.
 824  M=XGETF(ITCHD(1),IEXTD)
      N = IEXTS(KLOC)
C
C     LOOP WHICH LOOKS ALONG CHECK LINE
C     LOOK AT SQUARES IN DIRECTION OF CHECK
834   N = N+M
836   N1 = XMOVF(N)
C        LOOK AT BEARERS ON SQUARE
```

```
      DO 826 IDIR = 1,16
      IF (XABSF(IBEAR(N1,IDIR))-2) 826,826,827
 827  IF(XLBITF(IBEAR(N1,IDIR)+MCOL))826,828,826
C        SAME COLOR, MAY INTERPOSE OR CAPTURE CHECKER
C        IS IT PINNED
828   INTER = IBEAR(N1,IDIR)
      IF(IPIN(INTER))826,829,826
C        NOT PINNED
C             CONSTRUCT MOVE.  THERE ARE PAWN COMPLICATIONS.
829   IF(KIND(INTER)-1)830,831,830
C             A PAWN
831   IF(IDIR-4)832,832,833
C             VERTICAL DIRECTION.  OK IF SQUARE IS EMPTY.
832   IF(IOCC(N1)) 826, 8380, 826
C     IS THERE AN INTERVENING OCCUPIED SQUARE
8380  IF(XGETF(XMOVF(XGETF(LOC(INTER),IEXTS)+IEXTD(IDIR)),IOCC))
     1  826, 830, 826
C     DIAGONAL DIRECTION.  OK IF THE SQUARE IS OCCUPIED.
833   IF(IOCC(N1))830,826,830
C             CONSTRUCT MOVE.
830   K1 = K1 + 1
      MAVAIL(K1) = MSTO(INTER) + MS4M1(IDIR) + N1
826   CONTINUE
      IF (IOCC(N1)) 843,834,843
C             IF THE CHECKER IS A PAWN ANY EN PASSANT MOVES ARE OK
C     UNLESS THE MOVER IS PINNED.
 843  IF (XGETF(ITCH(1),KIND)-1) 825,840,825
 840  IF (NEP(NUMEP)-MOVENO) 825,844,825
 844  IF (XGETF(XMV3F(MEP1(NUMEP)),IPIN)) 845,841,845
841   K1 = K1+1
      MAVAIL(K1) = MEP1(NUMEP)
 845  IF (MEP2(NUMEP)) 846,825,846
 846  IF (XGETF(XMV3F(MEP2(NUMEP)),IPIN)) 825,842,825
842   K1 = K1 + 1
      MAVAIL(K1) = MEP2(NUMEP)
C        IF THERE ARE NO LEGAL MOVES IT IS MATE
 825  IF(K1)143,835,143
835   K1 = -1
143   NLOG = NLOG+1
      MLOG=MLOG+1
      LOGG(NLOG) = MIN
      IF(NLOG-100)144,145,145
145   WRITE TAPE 7,LOGG
      NLOG=0
144   RETURN
C
C     IS MOVE AN ENPASSANT CAPTURE,DOES IT ALLOW ONE, IS IT A PROMOTION
 400  IF(NFIRST(MOVER))402,402,412
 402  NFIRST(MOVER)=1
      NSPEC=NSPEC+1
      NUMB(NSPEC)=MOVENO
      NTYPE(NSPEC)=1
      IF (XTRANKF(MOVETO,MOVER)-4) 134,403,134
C     2ND RANK TO 4TH LOOK TO SIDES
```

```
403     DO 405 J=1,2
        IX=XMOVF(IEXTS(MOVETO)+IEXTD(2*J-1))
        IF(IX-64)404,404,405
 404    IY=IOCC(IX)
        IF(IY)405,405,407
 407    IF(KIND(IY)-1)405,408,405
 408    IF(XLBITF(IY+MOVER))405,405,409
C       THERE IS AN EN PASSANT TRY
409     IZ = IBEG(IY)+J-1
        IF (NEP(NUMEP)-MOVENO) 420,421,420
 420    NUMEP=NUMEP+1
        NEP(NUMEP)=MOVENO
        MEP1(NUMEP) = XABSF(MOVE(IZ))
        GO TO 405
421     MEP2(NUMEP) = XABSF(MOVE(IZ))
 405    CONTINUE
        GO TO 134
C            IS THIS MOVE A PROMOTION
412     IF(XADDF(MIN))419,418,419
C       NOT A PROMOTION.  IS IT AN EN PASSANT CAPTURE
 418    IF (MOVDIR-4) 134,134,413
 413    IF(IOCC(MOVETO))134,416,134
C       DIAGONAL MOVE TO EMPTY SQUARE
 416    IX=XMOVF(IEXTS(MQ)+XGETF(4-XABSF(13-MOVDIR-MOVDIR),IEXTD))
        NSPEC=NSPEC+1
        NUMB(NSPEC)=MOVENO
        NTYPE(NSPEC)=IX
        ICAPT(MOVENO)=-IOCC(IX)
        CALL PUTCH(IOCC(IX),0)
        GO TO 134
419     IPROM = XADDF(MIN)
        KIND(MOVER)=IPROM
        IF (XLBITF(MOVER)) 423,423,422
 422    MATW=MATW+KVAL(IPROM)-1
        GO TO 424
 423    MATB=MATB+KVAL(IPROM)-1
 424    NSPEC=NSPEC+1
        NUMB(NSPEC)=MOVENO
        NTYPE(NSPEC)=-1
        IEND(MOVER)=IBEG(MOVER)+NMOV(IPROM)-1
        GO TO 134
C
C       HANDLES FIRST MOVE OF KING AND
C       MAKES CASTLING MOVES
C
 460    IF(NFIRST(MOVER))134,462,134
 462    NFIRST(MOVER)=1
        NSPEC=NSPEC+1
        NUMB(NSPEC)=MOVE NO
        NTYPE(NSPEC)=1
C       TEST FOR CASTLING MOVE
        IF(XABSF(MOVETO-MQ)-2)134,463,134
463     IF(MOVETO-MQ)464,466,466
C       CASTLE QUEENS SIDE
```

```
 464   IA=-4+MQ
       JA=-1+MQ
       GO TO 467
C      CASTLE KINGS SIDE
 466   IA=3+MQ
       JA=1+MQ
 467   CALL PUTCH(MOVER,MOVETO)
 468   IAA=IOCC(IA)
       CALL PUTCH(IAA,JA)
       NTYPE(NSPEC)=-(IA-1+MSTO(IAA))
       GO TO 139
C
C      REVERT TAKES BACK MOVES
C
C
 150   IF (MOVEP(MOVENO)) 201,201,167
C      CHANGE SIDE TO MOVE
 201   MCOL=3-MCOL
       MIN=-0
       GO TO 165
C      NORMAL REVERSION
 167   MOVER=MOVEP(MOVENO)
       MOVETO=MOVEFR(MOVENO)
       ISPEC=0
       MIN = 0
C      IS THIS A SPECIAL MOVE
       IF(NUMB(NSPEC)-MOVENO)152,151,152
C      SPECIAL MOVE
 151   ISPEC=NTYPE(NSPEC)
       NUMB(NSPEC)=0
       NTYPE(NSPEC)=0
       NSPEC=NSPEC-1
C      SET UP VARIABLES
 152   MQ=LOC(MOVER)
       MCAPT = ICAPT(MOVENO)
       ICAPT(MOVENO) = 0
       MCOL = 2 -XLBITF(MOVER)
       KD=KIND(MOVER)
C      ORDINARY OR SPECIAL MOVE
       IF(ISPEC)153,154,154
C      SPECIAL,CASTLING OR PROMOTION
 153   IF(ISPEC+1)155,156,155
C      CASTLING
 155   MVR=XMV3F(ISPEC)
       NFIRST(MVR)=0
       NFIRST(MOVER)=0
       CALL PUTCH(MVR,XMV1F(ISPEC))
       GO TO 154
C
C      PROMOTION
 156   IF (XLBITF(MOVER)) 168,168,169
 169   MATW=MATW-XGETF(KIND(MOVER),KVAL)+1
       GO TO 170
 168   MATB=MATB-XGETF(KIND(MOVER),KVAL)+1
```

```
 170  KIND(MOVER)=1
C
C
C           WAS IT FIRST MOVE OF K, R, OR P
 154  IF (ISPEC-1) 171,163,171
C           RESTORE NFIRST
 163  NFIRST(MOVER)=0
C           MOVE PIECE BACK
 171  CALL PUTCH(MOVER,MOVETO)
C      WAS THE MOVE A CAPTURE OR EN PASSANT CAPTURE
       IF (MCAPT) 158,162,160
C      EN PASSANT CAPTURE
 158   CALL PUTCH(-MCAPT,ISPEC)
       GO TO 162
C      ORDINARY CAPTURE
 160   CALL PUTCH(MCAPT,MQ)
C
C      IS THERE AN EN PASSANT POSSIBILITY
 162   IF (NEP(NUMEP)-MOVENO) 165,166,165
C      YES, AT LEAST ONE
 166   NUMEP=NUMEP-1
       NEP(NUMEP+1)=0
       MEP1(NUMEP+1)=0
       MEP2(NUMEP+1)=0
C      RESET FUNCTIONS OF MOVENO
 165   MOVEP(MOVENO)=0
       MOVEFR(MOVENO)=0
       ICAPT(MOVENO)=0
       MOVENO=MOVENO-1
       GO TO 700
       END
```

```
*       LABEL
*       LIST8
        SUBROUTINE PUTCH (M6,M7)
C       DEC. 2, 1960, KOTOK, LIEBERMAN AND NIESSEN.
C
C       DIMENSION AND EQUIVALENCE STATEMENTS
        DIMENSION IOCC(64),LOC(32),NFIRST(22),NUMB(50),
       1NTYPE(50),IBEG(33),IEND(32),MOVE(504),ICAPT(150),
       2MOVEFR(150),MOVEP(150),JBEAR(1024),IBEAR(64,16),
       3KIND(32),MSVN(16),IPDIR(3,2),IEXTD(16),IEXTS(64),
       4M64M1(16),NMOV(6),IOPP(16)
        DIMENSION JPAWN(8)
        DIMENSION MSTO(32)
        DIMENSION MAVAIL(100),ITCH(2),ITCHD(2),IPIN(32)
        DIMENSION NEP(10),MEP1(10),MEP2(10)
        DIMENSION JPROM(4)
        DIMENSION LOGG(101)
        DIMENSION NZZZ(120)
        DIMENSION KVAL(6),KFORCE(64),KWORTH(64)
C               COMMON STATEMENTS
        COMMON IPDIR,IOPP,IEXTS,IEXTD,JPAWN,M64M1,MSVN,NMOV,MSTO,JPROM,
       1IBEAR,JBEAR,KIND,IEND,IBEG,IOCC,LOC,NFIRST,MOVE,IENDS,MOVEP,
       2MOVEFR,ICAPT,NUMB,NTYPE,ITCH,ITCHD,IPIN,NEP,MEP1,MEP2,LOGG,NLOG,
       3NZZZ,NUMTES,MAVAIL,IZ,IY,IX,IU,IT,ISPEC,IR,IQ,IPROM,IOPPD,INTER,
       4IDIR,ICHECK,IA,IAA,A,JA,JB,JC,JDIR,JD,JE,JF,JIN,JJ,JROOK,J,K1,KD,
       5K,L2,L,M4,MARET,MCAPT,MCOL,MIN,MOVDIR,MOVENO,MOVER,MOVETO,MQ,M,
       6MVR,N1,N2,NEWSQ,N,NSPEC,NUMEP,NPRINT,KIN,KVAL,KFORCE,KWORTH,MOBW,
       7MOBB,MATW,MATB
        EQUIVALENCE (IENDS,IBEG(33)),(NLOG,LOGG(101)),(NUMTES,NZZZ(120)),
       1(IBEAR,JBEAR)
        DIMENSION NUMBER(64)
        COMMON NUMBER
        COMMON MLOG
C       500 MOVES A PIECE FROM ONE SQUARE TO ANOTHER AND UPDATES THE
C         TABLES IBEAR, MOVE, LOC, IOCC, IBEG, IEND,.  IT USES 200, 300
C         AND 600 AS SUBROUTINES.
C
500     MVR = M6
        MTO = M7
        MOLDSQ = LOC(MVR)
        LOC(MVR)=MTO
C               IS MOVE FROM OFF BOARD
        IF (MOLDSQ) 503,523,503
C       ADD NEW PIECE TO MATERIEL COUNT
 523    IF (XLBITF(MVR)-1) 530,531,532
 532    STOP 532
 531    MATW=MATW+XGETF(KIND(MVR),KVAL)
        GO TO 516
 530    MATB=MATB+XGETF(KIND(MVR),KVAL)
C       A PIECE  COMING FROM OFF THE BOARD MAY NEED MOVE STORAGE
 516    IF(IBEG(MVR))506,517,506
 517    IOCC(MTO) = MVR
        K = KIND(MVR)
        IF(K-1)518,519,518
```

```
 518  MNREQ = NMOV(K)
      GO TO 600
 519  IF(XTRANKF(MTO,MVR)-7)518,520,518
 520  MNREQ = 56
      GO TO 600
C        DELETE OLD MOVES AND BEARINGS
 503  IOCC(MOLDSQ)=0
      M=IBEG(MVR)
      N=IEND(MVR)
      DO 501 J=M,N
      IF(MOVE(J))510,501,510
 510  K = XDELF(MOVE(J))
      IF (JBEAR(K+1)) 521,521,522
 522  L2=XLBITF(MVR)
      MOBW=MOBW-L2
      MOBB=MOBB+L2-1
 521  JBEAR(K+1)=0
      MOVE(J)=0
 501  CONTINUE
C                IS MOVE TO OFF BOARD
 502  IF (MTO) 506,524,506
 506  IOCC(MTO)=MVR
      IF(KIND(MVR)-1)512,513,512
C          IS THIS PAWN MOVING TO THE 7TH RANK
 513  IF(XTRANKF(MTO,MVR)-7)512,514,512
 514  IF (IEND(MVR)-IBEG(MVR)-55) 515,512,512
 515  MNREQ=56
      GO TO 600
C     UPDATE MOVES OF PIECE IN ALL DIRECTIONS.  DATUM IS MTOUP
512   MTOUP = MVR
 200  NOLDSQ=LOC(MTOUP)
      MSTOP = MSTO(MTOUP)
      K=KIND(MTOUP)
      GO TO (210,220,230,240,222,260),K
C
C                  ROOK IN ALL DIRECTIONS
 220  ASSIGN 221 TO JRET
      DO 221 IDIR=1,4
      L=IBEG(MTOUP)+MSVN(IDIR)-8
      GO TO 280
 221  CONTINUE
      GO TO 201
C
C                  BISHOP IN ALL DIRECTIONS
 240  ASSIGN 241 TO JRET
      DO 241 IDIR=5,8
      L=IBEG(MTOUP)+MSVN(IDIR)-36
      GO TO 280
 241  CONTINUE
      GO TO 201
C
C                  QUEEN IN ALL DIRECTIONS
 260  ASSIGN 261 TO JRET
      DO 261 IDIR=1,8
```

```
      L=IBEG(MTOUP)+MSVN(IDIR)-8
      GO TO 280
  261 CONTINUE
      GO TO 201
C
C     KING IN ALL DIRECTIONS
  222 N1=1
      GO TO 232
C                   N IN ALL DIRECTIONS
  230 N1=9
232   N2=N1+7
      L3=IBEG(MTOUP)-N1
      DO 271 IDIR=N1,N2
      L=L3+IDIR
C                   N IN GIVEN DIRECTION
C                   DATA ARE MTOUP, IDIR, NOLDSQ
  270 L1=M64M1(IDIR)+MSTOP
      NEWSQ=XMOVF(IEXTS(NOLDSQ)+IEXTD(IDIR))
C                   IS THE SQUARE ON THE BOARD
  273 IF(NEWSQ-64)272,272,271
C                   ON BOARD
  272 IF (IBEAR(NEWSQ,IDIR)) 279,279,268
  268 L10=XLBITF(IBEAR(NEWSQ,IDIR))
      MOBW=MOBW-L10
      MOBB=MOBB-1+L10
  279 L2=XLBITF(MTOUP)
      MOBW=MOBW+L2
      MOBB=MOBB-L2+1
  269 IBEAR(NEWSQ,IDIR)=MTOUP
C                   IS THE SQUARE OCCUPIED
  274 IF(IOCC(NEWSQ))275,276,277
  275 STOP275
C                   OCCUPIED.  IS THE COLOR THE SAME AS THAT OF THE MOVER
  277 IF(XLBITF(IOCC(NEWSQ)-MTOUP))276,278,276
  276 MOVE(L)=NEWSQ+L1
      GO TO 271
  278 MOVE(L)=-(NEWSQ+L1)
  271 CONTINUE
      GO TO 201
C
C     UPDATE MOVES OF PAWN IN ALL DIRECTIONS
C     210-217 AND 320-350
C     PURPOSE- TO UPDATE THE MOVES OF A PAWN IN ALL DIRECTIONS.
C     ASSIGNS ADDITIONAL STORAGE TO PAWNS REACHING THE 7TH RANK.
C     DOES NOT SET UP EN PASSANT MOVES.  USES 600, XLBITF, XMOVF,
C     XRANKF, IPDIR, NFIRST, IEXTS, IEXTD, IOCC,.
C     TABLES AFFECTED- MOVE, IBEG, IEND, IBEAR,.
C     LOCAL VARIABLES- J,K, L, JRET, MPREQ, MNREQ, K1 NEWSQ, IDIR, L1,
C     AND L2
C     DATA SUPPLIED - MTOUP, NOLDSQ, IENUS(INITIALLY)
210   K=XLBITF(MTOUP)+1
      L9 = IBEG(MTOUP)-1
      DO 211 J=1,3
      IDIR=IPDIR(J,K)
```

```
      L = L9+J
      ASSIGN 211 TO JARET
      GO TO 320
211   CONTINUE
      GO TO 201
 201  MSQ=MTO
      ASSIGN 508 TO MRET
      GO TO 300
C           IS MOVE FROM ON BOARD
C     REMOVE PIECE FROM MATERIEL COUNT
 524  IF (XLBITF(MVR)-1) 526,528,527
 526  MATB=MATB-XGETF(KIND(MVR),KVAL)
      GO TO 508
 527  STOP 527
 528  MATW=MATW-XGETF(KIND(MVR),KVAL)
 508  IF(MOLDSQ)511,509,511
 511  MSQ=MOLDSQ
      ASSIGN 509 TO MRET
      GO TO 300
509   RETURN
C                MOVE STORAGE CONTROL  600 TO 625
C           PURPOSE- TO EXPAND AND CONTRACT THE MOVE
C           STORAGE ALLOTTED TO PAWNS WHEN THEY
C           REACH THE 7TH RANK OR REVERT TO IT
C           TABLES AFFECTED-MOVE,IBEG,IEND
C     DATA SUPPLIED---MNREQ,MVR,IENUS(INITIALLY)
C         LOCAL VARIABLES M1,M,N,J6,K,M2
C
C                MOVE STORAGE CONTROL
 600  IF(504-IENUS-MNREQ)601,602,602
C                STORAGE AVAILABLE AT THE END
 602  IF (IBEG(MVR)) 604,604,605
C                MOVE THE MOVE INFORMATION
 605  M1=IENUS+1
      M=IBEG(MVR)
      N=IEND(MVR)
      DO 606 J6=M,N
      MOVE(M1)=MOVE(J6)
      MOVE(J6)=0
 606  M1=M1+1
 604  IBEG(MVR)=IENUS+1
      IENUS = IENUS + MNREQ
      IEND(MVR)=IENUS
      GO TO 512
C                NOT ENOUGH STORAGE, RESORT
C                MAKE SURE CAPTURED PIECES USE NO STORAGE
 601  DO 607 J6=1,32
      IF (LOC(J6)) 608,608,615
 608  IBEG(J6)=0
      IEND(J6) = 0
      GO TO 607
C          PAWNS ON OR BELOW 6TH RANK NEED ONLY 4 MOVES
 615  IF (XMINOF(1-KIND(J6),6-XTRANKF(LOC(J6),J6))) 607,616,616
 616  IEND(J6)=IBEG(J6)+3
```

```
 607  CONTINUE
      M1=1
 620  M2=0
      DO 609 J6=1,32
      IF(M1-IBEG(J6))612,611,609
C                HAS J ALREADY BEEN RE-ARRANGED
 612  IF(M2-IBEG(J6))613,617,617
 613  IF(M2)617,617,609
 617  M2=IBEG(J6)
      K=J6
      GO TO 609
C                NO NEED TO ARRANGE THESE MOVES
 611  M1=IEND(J6)+1
      GO TO 620
 609  CONTINUE
      IF(M2)622,622,623
C                RE-ARRANGE
 623  M=IBEG(K)
      N=IEND(K)
      IBEG(K)=M1
      DO 624 J6 = M,N
      MOVE(M1)=MOVE(J6)
      MOVE(J6)=0
 624  M1=M1+1
      IEND(K)=M1-1
      GO TO 620
C                STORAGE COMPLETELY RE-ARRANGED
 622  IENUS=M1-1
      IF(504-IENUS-MNREQ)625,602,602
C                TOTAL STORAGE TOO SMALL AFTER RE-ARRANGEMENT
 625  STOP 625
C
C     UPDATE ALL PIECES BEARING ON MSQ
 300  DO 301 IDIR=1,16
      IF (IBEAR(MSQ,IDIR)) 303,301,303
 303  MTOUP=XABSF(IBEAR(MSQ,IDIR))
      MSTOP = MSTO(MTOUP)
      K=KIND(MTOUP)
      NOLDSQ=LOC(MTOUP)
      ASSIGN 301 TO JRET
      GO TO (313,310,314,312,315,310),K
C     MOVE OF KNIGHT IN GIVEN DIRECTION
  314 N1=9
      GO TO 317
C     MOVE OF KING IN GIVEN DIRECTION
  315 N1=1
C     CHANGE LEGALITY OF KNIGHT OR KING MOVES
 317  I F (MVR-MTOUP) 311,301,311
 311  IF (XLBITF(MVR-MTOUP)) 301,316,301
  316 L=IBEG(MTOUP)+IDIR-N1
      MOVE(L)=-MOVE(L)
 301  CONTINUE
      GO TO MRET,(508,509)
C
```

```
C           UPDATE ROOK OR QUEEN IN GIVEN DIRECTION
 310  L=IBEG(MTOUP)+MSVN(IDIR)-8
      GO TO 280
C           UPDATE BISHOP IN GIVEN DIRECTION
 312  L=IBEG(MTOUP)+MSVN(IDIR)-36
      GO TO 280
 313  ASSIGN 301 TO JARET
      J=JPAWN(IDIR)
      L=IBEG(MTOUP)+J-1
      GO TO 320
C           UPDATE Q,B, OR R IN GIVEN DIRECTION
 280  L1 = M64M1(IDIR) + MSTOP
      L2=XLBITF(MTOUP)
      IQ=IEXTD(IDIR)
      IR=IEXTS(NOLDSQ)
      DO 281 J=1,7
      IR=IR+IQ
      NEWSQ=XMOVF(IR)
 288  IF(NEWSQ-64)284,284,283
 284  IF (IBEAR(NEWSQ,IDIR)) 282,282,299
 299  L10=XLBITF(IBEAR(NEWSQ,IDIR))
      MOBW=MOBW-L10
      MOBB=MOBB-1+L10
 282  MOBW=MOBW+L2
      MOBB=MOBB-L2+1
      IBEAR(NEWSQ,IDIR)=MTOUP
      J1=L+J
 289  IF(IOCC(NEWSQ))285,281,287
 285  STOP 2105
 281  MOVE(J1)=NEWSQ+L1
C           NON EXISTENT SQUARE
 283  GO TO JRET,(221,241,261,301)
C                       SQUARE OCCUPIED
 287  IF(XLBITF(IOCC(NEWSQ)-MTOUP))290,291,290
 290  MOVE(J1)=NEWSQ+L1
      GO TO 292
 291  MOVE(J1)=-(NEWSQ+L1)
 292  IF (J-6) 252,252,251
 252  DO 294 J3=J,6
      J1=L+J3+1
 293  IF(MOVE(J1))295,296,295
 296  GO TO JRET,(221,241,261,301)
 295  MOVE(J1)=0
      IR=IR+IQ
      NEWSQ=XMOVF(IR)
 286  IF (XABSF(IBEAR(NEWSQ,IDIR))-MTOUP) 294,298,294
 298  IBEAR(NEWSQ,IDIR)=0
      MOBW=MOBW-L2
      MOBB=MOBB+L2-1
 294  CONTINUE
 251  GO TO JRET,(221,241,261,301)
C     320 UPDATES A PAWN IN A GIVEN DIRECTION, COPIES MOVES OVER FOR A
C     PAWN ON THE 7TH RANK.
C     USES-XLBITF, XMOVF, IEXTS, IEXTD, M64M1, IOCC, NFIRST.
```

```
C       TABLES AFFECTED-IBEAR, MOVE.
C       LOCAL VARIABLES-(NEWSQ,L1,L2
C       DATA SUPPLIED-NOLDSQ,IDIR,MSTOP,MTOUP,JARET,L,J.
C
 320    NEWSQ=XMOVF(IEXTS(NOLDSQ)+IEXTD(IDIR))
        IF(NEWSQ-64)321,321,322
322     GO TO JARET,(211,301)
 321    L1 = M64M1(IDIR) + MSTOP
        L3=XLBITF(MTOUP)
        IF (IBEAR(NEWSQ,IDIR)) 342,342,343
 343    L10=XLBITF(IBEAR(NEWSQ,IDIR))
        MOBW=MOBW-L10
        MOBB=MOBB-1+L10
 342    IBEAR(NEWSQ,IDIR)=MTOUP
        MOVE(L)=NEWSQ+L1
        L2=IOCC(NEWSQ)
        IF(J-3)330,323,323
C                   MOVE IS DIAGONAL
 330    MOBW=MOBW+L3
        MOBB=MOBB-L3+1
        IF (L2) 328,328,326
 326    IF(XLBITF(L2+MTOUP))328,328,350
 328    MOVE(L)=-MOVE(L)
C                   PROMOTION POSSIBILITIES MAY HAVE BEEN SETUP
        IF (XTRANKF(NOLDSQ,MTOUP)-7) 338,353,338
C                   MOVE IS VERTICAL
 323    IBEAR(NEWSQ,IDIR)=-XABSF(IBEAR(NEWSQ,IDIR))
        IF (L2) 331,331,332
C                   CAN WE MOVE TWO SQUARES
 331    IF(NFIRST(MTOUP))334,334,335
 335    MOVE(L+1)=0
 350    IF (XTRANKF(NOLDSQ,MTOUP)-7) 338,353,338
C                   MAY BE ABLE TO MOVE TWO SQUARES
 334    NEWSQ=XMOVF(IEXTS(NEWSQ)+IEXTD(IDIR))
        IBEAR(NEWSQ,IDIR)=-MTOUP
        MOVE(L+1)=-XSIGNF(L1+NEWSQ,IOCC(NEWSQ)-1)
338     GO TO JARET,(211,301)
C                   REMOVE POSSIBLE FALSE BEARING
 332    MOVE(L)=-MOVE(L)
        MOVE(L+1)=0
        IF(NFIRST(MTOUP))350,339,350
 339    NEWSQ=NEWSQ+24-8*IDIR
        IF (IBEAR(NEWSQ,IDIR)) 338,341,338
341     IBEAR(NEWSQ,IDIR) = 0
        GO TO JARET,(211,301)
C             IF ON THE 7TH RANK MOVES MUST BE DUPLICATED
C              COPY MOVES
  353 MOVE(L+4)=MOVE(L)+XSIGNF(JPROM(2),MOVE(L))
        MOVE(L+8)=MOVE(L)+XSIGNF(JPROM(3),MOVE(L))
        MOVE(L+12)=MOVE(L)+XSIGNF(JPROM(4),MOVE(L))
        MOVE(L)=MOVE(L)+XSIGNF(JPROM(1),MOVE(L))
        GO TO JARET,(211,301)
C
          END
```

```
*       LABEL
*       LIST8
CONLINE CHESS MAIN PROGRAM, FEB. 28, 1962
        DIMENSION FOO(5000)
        DIMENSION LOC(32),NFIRST(22),KPAWNV(8),IEXTD(16),IEXTS(64)
        DIMENSION IPIN(32),IOPP(16),KIND(32),MAVAIL(100),KVAL(6)
        DIMENSION IHOPE(64),IEXCH(128)
        DIMENSION LISP(6000)
        COMMON FOO
        EQUIVALENCE(NSPEC,FOO(2913))
        EQUIVALENCE(FOO(2892),K1),(FOO(1463),KIND),(FOO(2765),MAVAIL)
        EQUIVALENCE(FOO(2900),MCOL)
        EQUIVALENCE(FOO(2703),IPIN),(FOO(1285),IOPP),(FOO(255),IBEAR)
        EQUIVALENCE(FOO(1365),IEXTD),(FOO(1301),IEXTS),(FOO(1527),IOCC)
        EQUIVALENCE(FOO(1591),LOC),(FOO(1623),NFIRST),(FOO(3003),KPAWNV)
        EQUIVALENCE (FOO(3121),PLY),(FOO(3120),IPE),(FOO(2917),KVAL)
        EQUIVALENCE (FOO(3051),MOBW),(FOO(3052),MOBB),(FOO(3123),IHOPE)
        EQUIVALENCE (FOO(9188),IEXCH),(FOO(3122),BACK),(FOO(3187),LISP)
        EQUIVALENCE (FOO(3053),MATW),(FOO(3054),MATB),(FOO(3119),MLOG)
        EQUIVALENCE (FOO(134),NLOG)
        EQUIVALENCE (FOO(2903),MOVENO)
        DIMENSION KPLY(20)
        EQUIVALENCE (KPLY, FOO(9167))
        EQUIVALENCE (FOO(9316), MOVES),(FOO(9317),NMOVES)
        CALL BEGIN
        READ 101, (KPLY(I), I = 1, 20)
101     FORMAT (2013)
26      J=1
        REWIND 6
        NMOVES=0
        CALL INITIA(J)
        CALL PRINT (-7)
        WRITE OUTPUT TAPE 100,1
1       FORMAT(59H0THE MIT CHESS PROGRAM WELCOMES YOU AS ITS WORTHY OPPONE
       1NT./117H IF YOU WISH TO PLAY WHITE, KEY IN THE NUMBER OF YOUR MOVE
       2 IN THE DECREMENT OF THE KEYS.  IF BLACK, SET KEYS TO ZERO./89H IF
       3 AT ANY TIME, YOU WISH TO START OVER, SET ADDRESS OF KEYS NON ZERO
       4.  THEN PRESS START./30H KEYS NEGATIVE PRINTS HISTORY./1H1)
        PAUSE
        IF(KEYS(J)) 3,3,2
3       IF(J) 4,4,5
4       WRITE OUTPUT TAPE 100,7
7       FORMAT(14H0MACHINE FIRST)
        GO TO 10
C
15      CALL REVERT
14      J=I
5       IF(K1-J)   69,8,8
8       J=J
        MOVES=MAVAIL(J)
        CALL UPDATE(MAVAIL(J))
        CALL PRINT (-7)
10      WRITE OUTPUT TAPE 100,' 9
9       FORMAT(95H0IF THIS MOVE IS CORRECT, SET KEYS TO ZERO AND PRESS STA
```

```
     1RT. OTHERWISE SET KEYS TO CORRECT MOVE./1H1)
1003  PAUSE
      IF(KEYS(I)) 1002,11,2
11    IF (I) 12,12,13
13    IF (J) 14,14,15
12    IF (K1)    16,16,18
16    WRITE OUTPUT TAPE 100, 19
19    FORMAT(6HODARN./41H1CARE TO TRY AGAIN...  PRESS START IF SO./1H1)
      PAUSE
      GO TO 2
18    L = XTIMEF(L)
      CALL TREE (MOVE)
      TIME = XLAPSEF(L)
      CALL UPDATE(MOVE)
B     CALL PRINT (407777000000)
   33 IF(K1)20,16,17
20    WRITE OUTPUT TAPE 100, 21
21    FORMAT(16HOWHOOPEE, I WIN./43H1 CARE TO LOSE AGAIN...  PRESS START
     1 IF SO./1H1)
      PAUSE
      GO TO 2
17    WRITE OUTPUT TAPE 100, 22, TIME
22    FORMAT (24HOTHE PRECEDING MOVE TOOK, -1PF4.1, 9H MINUTES./42HOPLEA
     1SE KEY IN YOUR REPLY AND PRESS START.)
      REWIND 7
      NLOG = 0
      MLOG = 0
25    PAUSE
      IF(KEYS(J)) 69,23,2
23    IF(J) 69,69,5
C     ERROR PSEUDO STOP
69    WRITE OUTPUT TAPE 100,691
691   FORMAT(25H1ILLEGAL MOVE, TRY AGAIN./1H1)
      GO TO 25
C     START OVER
2     IF (SENSE SWITCH 3) 709, 7090
7090  BACKSPACE 4
      BACKSPACE 4
B709  CALL PRINT (777400000)
      REWIND 7
      MLOG = 0
      NLOG = 0
      GO TO 26
B1002 CALL PRINT (410000000000)
      GO TO 1003
      END
```

```
*       LABEL
*       FAP
        COUNT   354
*       FUNCTION INITIA, M179 CHESS, APR. 17, 1961
        ENTRY   INITIA
INITIA  SXD     XR4,4                   INITIALIZE
        SXA     XR2,2
        SXA     XR1,1
        STI     INDIC
        CLA*    1,4
        TZE     A1342
        AXT     32,1
LP32    STZ     IBEG+1,1
        STZ     IEND+1,1
        STZ     LOC+1,1
        STZ     IPIN+1,1
        STZ     LOCIN+1,1
        TIX     LP32,1,1
        AXT     100,1
LP100   STZ     MAVAIL+1,1              CLEAR TABLES
        TIX     LP100,1,1
        STZ     IENUS
        AXT     22,1
LP22    STZ     NFIRST+1,1
        TIX     LP22,1,1
        AXT     50,1
LP50    STZ     NUMB+1,1
        STZ     NTYPE+1,1
        TIX     LP50,1,1
        AXT     64,1
LP64    STZ     IOCC+1,1
        PXD     ,1
        STO     NUMBER+1,1
        TIX     LP64,1,1
        AXT     504,1
LP504   STZ     MOVE+1,1
        TIX     LP504,1,1
        AXT     150,1
LP150   STZ     ICAPT+1,1
        STZ     MOVEFR+1,1
        STZ     MOVEP+1,1
        TIX     LP150,1,1
        STZ     MATW
        STZ     MATB
        STZ     MOBW
        STZ     MOBB
        STZ     NUMEP
        STZ     ISPEC
        STZ     NSPEC
        STZ     MOVENO
        AXT     1024,1
LP1024  STZ     JBEAR+1,1
        TIX     LP1024,1,1
        STZ     ITCH
```

```
        STZ     ITCH-1
        STZ     ITCHD
        STZ     ITCHD-1
        AXT     10,1
LP10    STZ     NEP+1,1
        STZ     MEP1+1,1
        STZ     MEP2+1,1
        TIX     LP10,1,1
        CLA     =1B17
        AXT     7,1
LP722   STO     KIND+1,1
        TXI     *+1,1,1
        TXL     LP722,1,22
INPUT   CLA     =1B17           READ PROBLEM
        STO     LOC1
        STO     COLOR
        AXT     INS+1,4
        SXA     INS,4
        STZ     LETTER
        AXT     0,2
CARD    CAL     =4B17           READ IN ANOTHER CARD
        TSX     $(TSH),4
        PZE     =H(12A6)
        AXT     12,1            FORTRAN READ INPUT TAPE 4
        STR
        STQ     TABLE+12,1
        TIX     *-2,1,1
        TSX     $(RTN),4
        CAL     TABLE
        LAS     =HFORTRA
        TRA     *+2
        TRA     B1234
        AXT     12,1            WORD COUNT
B       AXT     6,4             CHARACTER COUNT
        LDQ     TABLE+12,1
A       SXA     CHLOOP,4
        PXD
        LGL     6
INS     TRA     *
        CAS     =H00000.
        TRA     *+2
        TRA     PERIOD
        CAS     =H00000         BLANK
        TRA     *+2
        TRA     CHLOOP          BLANKS IGNORED
        CAS     =H00000*
        TRA     *+2
        TRA     COLOR1
        CAS     =H000009        NUMERAL
        TRA     *+3
NOP     NOP
        TRA     NUMBUH
        CAS     =H000000(       OPEN PARENTHESIS
        TRA     *+2
```

```
        TRA     OPEN
        CAS     =H00000)
        TRA     *+2
        TRA     CLOSE
        CAS     =H00000Q            Q OR K BEGINS A NEW PIECE
        TRA     *+2
        TRA     BREAK
        CAS     =H00000K
        TRA     *+2
        TRA     BREAK
        CAS     =H00000/
        TRA     *+2
        TRA     COMENT
        ADD     LETTER              ANYTHING ELSE ASSUMED LETTER
SHIFT   ALS     6
        STO     LETTER
        TXI     CHLOOP,2,1          INCREASE LETTER COUNT
COLOR1  TSX     LOOKUP,4
        STZ     COLOR
        TRA     RESETL
B1234   CAL     =4B17
        TSX     $(BST),4
        PXD
        LXD     XR4,4
        TRA     A1342
NUMBUH  STO     NUM
        TSX     LOOKUP,4
        CLA     NUM
        ALS     18
        ADD     LOC1
        CAS     =65B17
        TSX     ERROR,4
        NOP
        STO     LOC1
RESETL  AXT     0,2                 RESET CHARACTER COUNTER
        STZ     LETTER
        AXT     INS+1,4
        SXA     INS,4
CHLOOP  AXT     **,4
        TIX     A,4,1
        TIX     B,1,1
        TRA     CARD                READ ANOTHER CARD
LOOKUP  CLA     LETTER              CLOSED SUBROUTINE TO LOOKUP PIECE
        TXL     FOUND1,2,0
        TXL     ONE,2,1
        TXL     TWO,2,2
        TXL     THREE,2,3
        TSX     ERROR,4
THREE   ALS     12
        TRA     PLACE               NORMALIZE
TWO     ALS     18
        ORA     =H00 000
        TRA     PLACE
ONE     ALS     18
```

```
        ORA     =H 0 000
PLACE   ORA     =H000---
        ZET     COLOR
        ORA     =H000
        AXT     32,2
        LAS     PIECES,2
        TRA     *+2
        TRA     FOUND
        TIX     *-3,2,1
        TSX     ERROR,4
FOUND   CLA     LOC1                    INCREMENT LOCATION COUNTER
        CAS     =65B17
        NOP
        TSX     ERROR,4
        ZET     LOCIN+1,2
        TSX     ERROR,4
        STO     LOCIN+1,2
        ADD     =1B17
        STO     LOC1
        STL     COLOR
        TXH     FOUND1,2,22
        CLA     LOCBEG+1,2              SET UP NFIRST TABLE
        SUB     LOCIN+1,2
        TZE     *+2
        CLA     =1B17
        SSP
        STO     NFIRST+1,2
FOUND1  TRA     1,4
COMENT  AXT     COMEN1,4
        TRA     CHLOOP-1
COMEN1  CAS     =H00000/
        TRA     CHLOOP
        TRA     RESETL
        TRA     CHLOOP
OPEN    CLA     LETTER
        STO     CHANGE
        SXA     MOVED+1,2
        TNZ     RESETL
        TSX     ERROR,4
CLOSE   NZT     LETTER
        TSX     ERROR,4
        CLA     LETTER
        CAS     =H0000MO
        TRA     *+2
        TRA     MOVED
        TSX     LOOKUP,4
CLOSE1  CLA     CHANGE                  PROMOTED PIECE HANDLED HERE
        RIL     7
        CAS     =H0000RO
        TRA     *+2
        LDI     =2B17
        CAS     =H0000BO
        TRA     *+2
        LDI     =4B17
```

```
         CAS     =H0000N0
         TRA     *+2
         LDI     =3B17
         CAS     =H0000Q0
         TRA     *+2
         LDI     =6B17
         LFT     7
         TRA     CLOSE2
         TSX     ERROR,4
CLOSE2   TXH     *+2,2,6
         TSX     ERROR,4
         STI     KIND+1,2
         TRA     MOVED1
MOVED    CLA     CHANGE          (M) MEANS PIECE HAS MOVED
         AXT     **,2
         STO     LETTER
         TSX     LOOKUP,4
MOVED1   TXL     *+2,2,22
         TSX     ERROR,4
         CLA     =1B17
         STO     NFIRST+1,2
         TRA     RESETL
ERROR    SXA     ERLOC,4
         STL     COLOR
         LAC     ERLOC,4
ERROR1   TIX     *+1,4,INITIA-9
         SXA     ERLOC,4
         LXD     XR4,4
         CLA*    1,4
         STO     J
         CAL     =100B17
         TSX     $(STH),4
         PZE     ERFOR
         LDQ     ERLOC
         STR
         LDQ     J
         STR
         LDQ     LOC1
         STR
         AXT     12,2
         LDQ     TABLE+12,2
         STR
         TIX     *-2,2,1
         TSX     $(FIL),4
         NZT     COLOR
         TRA     A5678
         AXT     ERROR3,4
         TRA     CHLOOP-1
ERROR3   CAS     =H00000.
         TRA     CHLOOP
         TRA     *+2             LOOK FOR END OF PROBLEM
         TRA     CHLOOP
A5678    LXD     XR4,4
XR4      SYN     INITIA-2
```

```
         CLA*     1,4
         SUB      =1B17
         STO*     1,4
         TNZ      LP32-1
         TRA      XR1-2
BREAK    STO      KORQ
         TSX      LOOKUP,4
         AXT      0,2
         CLA      KORQ
         TRA      SHIFT
ERROR2   STZ      COLOR
         TRA      ERROR1
PERIOD   TSX      LOOKUP,4
         CLA      LOC1
         SUB      =65B17
         AXT      *+1,4
         TNZ      ERROR2
         CLA      =2B17
         ZET      COLOR
         SUB      =1B17
         STO      MCOL
         AXT      1,1
PTCH     NZT      LOCIN+1,1
         TRA      PTCHLP
         SXD      JIN,1
         PXA      LOCIN+1,1
         SUB      *-1
         STA      *+3
         CALL     PUTCH,JIN,LOCIN
PTCHLP   TXI      *+1,1,1
         TXL      PTCH,1,32
         CALL     SETUP
         LXD      XR4,4
         CLA*     1,4
A1342    SUB      =1B17
         STO*     1,4
XR1      AXT      **,1
XR2      AXT      **,2
         LDI      INDIC
         TRA      2,4
         BCI      1,6)
         BCI      1,1H012A
         BCI      1,CARD./
         BCI      1,OWING
         BCI      1,N FULL
         BCI      1,OUND O
         BCI      1,RROR F
         BCI      1,32H. E
         BCI      1,C1=14,
         BCI      1,7H, LO
         BCI      1, J=14,
         BCI      1,ATIVE.
         BCI      1,3H REL
         BCI      1,ONO6,1
```

```
        BCI     1,LOCATI
        BCI     1,IA AT
        BCI     1,Y INIT
        BCI     1,OUND B
        BCI     1,RROR F
ERFOR   BCI     1,(34H4E
ZILCH   COMMON  12561
R       COMMON  1
*       TEMPORARY STORAGE
J       PZE
COLOR   PZE
INDIC   PZE
ERLOC   PZE
CHANGE  PZE
LETTER  PZE
LOC1    PZE
NUM     PZE
KORQ    PZE
JIN     PZE
TABLE   BSS     12
        BSS     31
LOCIN   BSS     1
ITCH    SYN     R+9863
ITCHD   SYN     R+9861
IBEG    SYN     R+12561
IEND    SYN     R+11067
LOC     SYN     R+10971
IPIN    SYN     R+9859
MAVAIL  SYN     R+9797
IENUS   SYN     R+12529
NFIRST  SYN     R+10939
NUMB    SYN     R+9963
NTYPE   SYN     R+9913
IOCC    SYN     R+11035
NUMBER  SYN     R+9507
MOVE    SYN     R+10917
ICAPT   SYN     R+10113
MOVEFR  SYN     R+10263
MOVEP   SYN     R+10413
JBEAR   SYN     R+12307
NEP     SYN     R+9827
MEP1    SYN     R+9817
MEP2    SYN     R+9807
KIND    SYN     R+11099
MCOL    SYN     R+9662
PIECES  SYN     R+9624
LOCBEG  SYN     R+9581
MOVENO  SYN     R+9659
MATB    SYN     R+9508
MATW    SYN     R+9509
MOBB    SYN     R+9510
MOBW    SYN     R+9511
NUMEP   SYN     R+9648
ISPEC   SYN     R+9692
```

```
NSPEC   SYN       R+9649
        END
```

```
*       LABEL
*       LIST8
C           CHESS PRINT TABLE ROUTINE
        SUBROUTINE PRINT (CODE)
C
C       CONTROL WORD BITS ARE IN DECREMENT
C         1 PRINTS NUMBER, MOVER, MOVETO ON-LINE, OTHERWISE OFF-LINE.
C         2 PRINTS BOARD ON-LINE, OTHERWISE OFF-LINE.
C         4 PRINTS MAVAIL, ON-LINE IF CONTROL WORD IS NEGATIVE.
C        10 PRINTS MAT, MOB, COLOR, MOVENO, NSPEC, ICHECK, MLOG OFFLINE.
C        20 PRINTS LOC, IBEG, IEND, NFIRST, KIND, IPIN OFF-LINE.
C        40 PRINTS MOVEP, MOVEFR, ICAPT OFF-LINE.
C       100 PRINTS NUMB, ITCH, ITCHD, NEP, MEP1, MEP2 OFF-LINE.
C       200 PRINTS LOG OFF-LINE.
C       400 PRINTS IBEAR OFF-LINE.
C       1000 PRINTS MOVE TABLE OFF-LINE.
C       2000 PRINTS PRINCIPAL VARIATION, ONLINE IF NEGATIVE
C       4000 PRINTS MOVE TREE OFF LINE
C       10000 PRINTS HISTORY , ONLINE IF NEGATIVE
C       DIMENSION AND EQUIVALENCE STATEMENTS
        DIMENSION IOCC(64),LOC(32),NFIRST(22),NUMB(50),
       1NTYPE(50),IBEG(33),IEND(32),MOVE(504),ICAPT(150),
       2MOVEFR(150),MOVEP(150),JBEAR(1024),IBEAR(64,16),
       3KIND(32),MSVN(16),IPDIR(3,2),IEXTD(16),IEXTS(64),
       4M64M1(16),NMOV(6),IOPP(16)
        DIMENSION JPAWN(8)
        DIMENSION MSTO(32)
        DIMENSION MAVAIL(100),ITCH(2),ITCHD(2),IPIN(32)
        DIMENSION NEP(10),MEP1(10),MEP2(10)
        DIMENSION JPROM(4)
        DIMENSION LOGG(101)
        DIMENSION NZZZ(120)
        DIMENSION KVAL(6),KFORCE(64),KWORTH(64)
C               COMMON STATEMENTS
        COMMON IPDIR,IOPP,IEXTS,IEXTD,JPAWN,M64M1,MSVN,NMOV,MSTO,JPROM,
       1IBEAR,JBEAR,KIND,IEND,IBEG,IOCC,LOC,NFIRST,MOVE,IENUS,MOVEP,
       2MOVEFR,ICAPT,NUMB,NTYPE,ITCH,ITCHD,IPIN,NEP,MEP1,MEP2,LOGG,NLOG,
       3NZZZ,NUMTES,MAVAIL,IZ,IY,IX,IU,IT,ISPEC,IR,IQ,IPROM,IOPPD,INTER,
       4IDIR,ICHECK,IA,IAA,A,JA,JB,JC,JDIR,JD,JE,JF,JIN,JJ,JROOK,J,K1,KD,
       5K,L2,L,M4,MARET,MCAPT,MCOL,MIN,MOVDIR,MOVENO,MOVER,MOVETO,MQ,M,
       6MVR,N1,N2,NEWSQ,N,NSPEC,NUMEP,NPRINT,KIN,KVAL,KFORCE,KWORTH,MOBW,
       7MOBB,MATW,MATB
        EQUIVALENCE (IENUS,IBEG(33)),(NLOG,LOGG(101)),(NUMTES,NZZZ(120)),
       1(IBEAR,JBEAR)
        DIMENSION NUMBER(64), IEXCH(128)
        COMMON NUMBER
        COMMON MLOG
        DIMENSION LISP (6000), IHOPE (64)
        COMMON IPE, PLY, BACK, IHOPE, LISP, IPRINT
        COMMON IEXCH
        COMMON MOVES,NMOVES
        DIMENSION M1(100), M2(100), AM1(100), AM2(100)
        EQUIVALENCE (M1, AM1), (M2, AM2)
        EQUIVALENCE (I,A1)
```

```
C
      CODEWD=CODE
      IPRINT = IPRINT+1
C
C     NUMBER, MOVER, MOVETO
B     IF (CODEWD*000001000000) 6969, 1000, 1001
 1000 N=2
      GO TO 5
 1001 N = 100
    5 CALL JUNPAK( MOVETO+MSTO (MOVER) -1, M1 (1), M1 (2))
      WRITE OUTPUT TAPE N,910, IPRINT, M1 (1), M1 (2)
  910 FORMAT (21H1SET OF TABLES NUMBER,I3,10H - MOVE IS ,2A6)
C
C              IOCC
B      IF (CODEWD*000002000000) 6969, 47, 48
   47 N = 2
      GO TO 49
   48 N = 100
   49 CALL BOARD (N)
C
C             MAVAIL
B     IF (CODEWD*000004000000) 6969, 130, 50
   50 IF (CODEWD) 51, 6969, 52
   51 N = 100
      GO TO 54
   52 N = 2
54    IF(K1)45,42,44
   42 WRITE OUTPUT TAPE N, 70
   70 FORMAT (10H STALEMATE )
      GO TO 475
   45 WRITE OUTPUT TAPE N, 73
   73 FORMAT (10H CHECKMATE )
      GO TO 475
   44 WRITE OUTPUT TAPE N,980
  980 FORMAT (7H MAVAIL )
82    DO 17 I=1,K1
   17 CALL JUNPAK (MAVAIL (I), M1 (I), M2 (I))
      WRITE OUTPUT TAPE N, 1391, (M1(I), M2(I), I=1,K1)
  475 WRITE OUTPUT TAPE N, 139
  139 FORMAT (1H4)
1391  FORMAT (1H0,20A6)
130   CONTINUE
C
C     2000 PRINTS PRINCIPAL VARIATION, ONLINE IF NEG.
B     IF(CODEWD*002000000000) 6969,224,223
223   N=2
      IF(CODEWD) 221,220,220
221   N=100
220   I99=1
      CALL JUNPAK (MOVES,M1(1),M1(51))
      I=1
230   INT=XBANDF(XADDF(LISP(I+1)),127)
      IF(INT) 215,215,225
225   INT=I+INT+1
```

```
      I99=I99+1
      M1(I99)=XDECF(LISP(INT))+XSHIFTF(XTAGF(LISP(INT)),-18)
      CALL JUNPAK(M1(I99),M1(I99),M1(I99+50))
      I=XADDF(LISP(INT))
      GO TO 230
215   WRITE TAPE 6,I99,M1
      NMOVES=NMOVES+1
      WRITE OUTPUT TAPE N, 222, LISP(I+1),MLOG,(M1(I),M1(I+50),I=2,I99)
222   FORMAT (21H1PRINCIPAL VARIATION ///H VALUE=,I7,8H EFFORT=,I7/
     1(1H0,20A6))
224   CONTINUE
C
C     4000 PRINTS MOVE TREE
B     IF(CODEWD*004000000000) 6969,270,261
B261  AM1(1)=3
      LEVEL=1
      PRINT 260,(I,I=1,20)
260   FORMAT (1H1,51X,13HTHE MOVE TREE/6HOLEVEL,2015,8H   VALUE)
      I=3
263   IF(LISP(I)) 262,270,264
264   I=I+1
      GOTO 263
B262  AM1(LEVEL)=(AM1(LEVEL)*77777)+AI
      CALL WRITE(LISP(I),LEVEL)
      I=XADDF(LISP(I))
      IF(I) 270,269,271
271   IF(LISP(I)) 268,270,265
265   I=I+2
      LEVEL=LEVEL+1
      M1(LEVEL)=XSHIFTF(I,-18)
      GO TO 263
268   PRINT 274,LISP(I+1)
274   FORMAT (1H+,105X,110)
269   I=XDECF(M1(LEVEL))-1
      IF (XADDF(M1(LEVEL))-1) 262,262,267
267   LEVEL=LEVEL-1
      IF(LEVEL)270,270,269
270   CONTINUE
C
C          EVALUATION PARAMETERS
B     IF(CODEWD*000010000000) 6969,60,134
  134 PRINT 133, MATW, MOBW, MATB, MOBB
133   FORMAT (1H2,8X,19H MATERIEL  MOBILITY/6H WHITE,2I10/6H BLACK,2I10)
      IF (MCOL-1) 6969, 20, 21
B   20 AM2 = 606630316325
      GO TO 62
B   21 AM2 = 602243212342
   62 PRINT 22, K1, M2 (1), MOVENO, NSPEC, ICHECK, MLOG
   22 FORMAT(19H NUMBER OF MOVES = I3/8H MCOL IS A6/10H MOVENO = I3/9H N
     1SPEC = I4/9H ICHECK =I4//H MLOG =I6)
60    CONTINUE
C
C               PRINT THE OTHER TABLES
C     LOC, IBEG, IEND, NFIRST, KIND, IPIN
```

```
B       IF(CODEWD*000020000000) 6969,80,63
63      WRITE OUTPUT TAPE 2,2,(I,I=1,32),(LOC(I),I=1,32),(IBEG(I),I=1,32),
       1(IEND(I),I=1,32),(NFIRST(I),I=1,22),(KIND(I),I=1,32),(IPIN(I),I=1,
       232)
     2 FORMAT (8HOPIECE  32I3/8H LOC    32I3/8H IBEG    32I3/8H IEND
       132I3/8H NFIRST 22I3/8H KIND   32I3/8H IPIN   32I3)
80      CONTINUE
C
C       MOVEP, MOVEFR, ICAPT
B       IF(CODEWD*000040000000) 6969,90,81
    81  WRITE OUTPUT TAPE 2,8,(MOVEP(I),I=1,MOVENO)
     8  FORMAT (6H MOVEP19I6/(20I6))
        WRITE OUTPUT TAPE 2,7,(MOVEFR(I),I=1,MOVENO)
     7  FORMAT (6H MOVFR19I6/(20I6))
        WRITE OUTPUT TAPE 2,6,(ICAPT(I),I=1,MOVENO)
     6  FORMAT (6H ICAPT19I6/(20I6))
90      CONTINUE
C
C       NUMB, NTYPE, ITCH, ITCHD, NEP, MEP1, MEP2
B       IF(CODEWD*000100000000) 6969,162,95
    95  WRITE OUTPUT TAPE 2,91,(NUMB(I),I=1,NSPEC)
    91  FORMAT (6H NUMB 15I6/(20I6))
        WRITE OUTPUT TAPE 2,92,(NTYPE(I),I=1,NSPEC)
92      FORMAT(6H NTYPE15I6/(20I6))
        WRITE OUTPUT TAPE  2,93,(ITCH(I),I=1,2),(ITCHD(I),I=1,2)
 93     FORMAT (6H ITCH 2I3,8H  ITCHD 2I3)
C
C            SET UP NEP, MEP1, AND MEP2 FOR OUTPUT
       DO 153 J=1,60
 153   M1(J)=0
       DO 150 I=1,10
       IF (NEP(I)) 151,150,151
 151   M1(I)=XMV3F(MEP1(I))
       M1(I+20)=XMV2F(MEP1(I))
       M1(I+40)=XMV1F(MEP1(I))
       IF (MEP2(I)) 155,150,155
 155   M1(I+10)=XMV3F(MEP2(I))
       M1(I+30)=XMV2F(MEP2(I))
       M1(I+50)=XMV1F(MEP2(I))
 150   CONTINUE
C            PRINT OUT THE EN PASSANT TABLES
       WRITE OUTPUT TAPE 2,154,(NEP(I),I=1,10),(M1(I),I=1,60)
 154   FORMAT (4H NEP10I3,5H MEP110I3,5H MEP210I3/(I42,9I3,I8,9I3))
162    CONTINUE
C
C            WRITE THE LOG
B       IF(CODEWD*000200000000) 6969,170,164
  164   PRINT 165, MLOG
  165   FORMAT (17H1 THE LOG---MLOG=,I5//)
        I1=0
        IF (MLOG-100) 160,160,161
 161    REWIND 7
        DO 166 I3=100,MLOG,100
        I1=I3
```

```
      READ TAPE 7,M1
      DO 1640 I=1, 100
 1640 CALL JUNPAK (M1 (I), M1 (I), M2 (I))
  166 PRINT 163, (M1 (I), M2 (I), I=1, 100)
163   FORMAT (1H0,2A6,A7,A6,A7,A6,A7,A6,A7,A6,A7,A6,A7,A6,A7,A6,A7,A6,
     1A7,A6)
 160  I2=MLOG-I1
      IF (I2) 170, 170, 167
 167  DO 169 I=1,I2
  169 CALL JUNPAK (LOGG (I), M1(I), M2 (I))
      PRINT 163, (M1 (I), M2 (I), I=1,I2)
170   CONTINUE
C
C           IBEAR
B     IF(CODEWD*000400000000) 6969,200,168
  168 PRINT 10, ((I, I=1, 16), J=1, 2), (I, (IBEAR (I, J), J=1, 16),
     1NUMBER (I+32), (IBEAR (I+32, J), J=1, 16) I=1, 32)
 10   FORMAT (6H1IBEAR/16,15I3,116,15I3/(17I3,I13,16I3))
200   CONTINUE
C
C           MOVE
B     IF(CODEWD*001000000000) 6969,201,210
  210 PRINT 94
   94 FORMAT (12H0MOVE TABLE.)
      DO 11 I=1,32
      IF(LOC(I))12,11,12
 12   M=IBEG(I)
      N=IEND(I)
      DO 13 J=M,N
      K=J-M+1
      IF (MOVE (J)) 110, 111, 110
 111  M1(K)=0
      M2(K)=0
      GO TO 13
  110 M1 (K)=XSIGNF (XMV1F (MOVE (J)), MOVE (J))
      M2 (K)=XMV2F (MOVE (J))
 13   CONTINUE
      K3=XMINOF(28,N-M+1)
      WRITE OUTPUT TAPE 2,15,I,(M1(L),L=1,K3)
15    FORMAT (23H MOVES OF PIECE NUMBER ,I2/(1H0,2814))
      WRITE OUTPUT TAPE 2,16,(M2(L),L=1,K3)
16    FORMAT (1H0,2814)
      IF(N-M+1-28)11,11,113
 113  K3=N-M+1
      WRITE OUTPUT TAPE 2,16,(M1(L),L=29,K3)
      WRITE OUTPUT TAPE 2,16,(M2(L),L=29,K3)
 11   CONTINUE
201   CONTINUE
C
C     10000 PRINTS HISTORY
B     IF(CODEWD*010000000000) 6969,350,310
310   N=2
      IF(CODEWD) 311,312,312
311   N=100
```

```
312    IF(NMOVES) 350,350,313
313    REWIND 6
       WRITE OUTPUT TAPE N, 322
322    FORMAT (31HILEVEL    OPPONENT       MACHINE,10X,19HPRINCIPAL VARIAT
      1ION)
       DO 320 I98=1,NMOVES
       READ TAPE 6,I99,M1
320    WRITE OUTPUT TAPE N, 321,I98,(M1(I),M1(I+50),I=1,I99)
321    FORMAT (1H0,I5,2(2X,2A6),2X,14A6/(36X,14A6))
350    CONTINUE
C
  600 RETURN
 6969 PRINT 6970
 6970 FORMAT (47HOLOSE. LOGIC OF PROGRAM MAKES THIS IMPOSSIBLE. )
       GO TO 600
       END
```

```
        END
*        LABEL
*        FAP
         COUNT   55
*        FTNBOL BINARY LOADER
*        LOADS COLUMN ABSOLUTE FROM TAPE A2.
         REM 0056 SYM. CARDS DIST.    535 RCV. 12-03-58CORR. OF DIST.52711
*   WD BTU2, BINARY TAPE UPPER LOADER
*
         ENTRY   FTNBOL
L        TAPENO  A2                      INPUT TAPE
FTNBOL   TEFL    *+1
         SXA     TR2,1
         SXA     TR2+1,2
 AXT     AXT     1,2
 CLEAR   CLM
         RTBL
         RCHL    IOCT
         LCHL    TXH
         TEFL    TR3
         LDQ     CW
         TQP     *+2
TR3      CALL    EXIT
         LGL     6
         ALS     3
         LGL     6
         ARS     3
         LGL     12
         SLW     READ
         RCHL    READ
         STA     TR1
 PDC     LDC     READ,1
         STQ     READ
TRAN     TNX     TR2,1
         CLA     CW
         LGR     12
         TCOL    *
         TXI     *+1,1,1
TR1      ACL     **,1
 TXH     TXH     *-2,1
 FOLD    LDQ     EOF
         LGR     24
         ALS     24
         STQ     CW
         ACL     CW
         ZET     CW
         TRA     FOLD
         TRCL    NG
         ERA     READ
         ZET     READ
         TNZ     NG
         TRA     AXT
NG       TIX     TR3,2,2
         BSRL
```

```
        TXI     CLEAR,2,1
READ    PZE
 IOCT   IOCT    CW,0,1
 EOF    HTR     AXT
CW      PZE
TR2     AXT     **,1
        AXT     **,2
        TRA     1,4
        END
```

63

```
*       LABEL
*       FAP
        COUNT   270
*MISPX  BUGGERED VERSION OF MISPH- (SPH),(SPHM),(STH),(STHM),(SCH),
*        AND (SCHM).   THIS VERSION RECOGNIZES TAPE 100 AS MEANING
*        WRITE ON TAPE 2, AND PRINT ON LINE.
        ENTRY   (SPH)
        ENTRY   (SPHM)
        ENTRY   (STH)
        ENTRY   (STHM)
        ENTRY   (STHD)
        ENTRY   (SCH)
        ENTRY   (SCHM)
        REM
(PRCT)  EQU     88
(PUCT)  EQU     89
(ELCT)  EQU     90
LNCNT.  EQU     97
PUNSW.  EQU     4               ON LINE PUNCH SWITCH
PRNSW.  EQU     5               ON LINE PRINT SWITCH
        REM
(SPHM)  CAL     =02000000       (SPHM)=WRITE OUTPUT TAPE 2
(STHM)  STL     MONSW.          SET SWITCH FOR MONITOR CONTROL
        CAS     =100B17         CHECK FOR TAPE 100
        TRA     *+2
        TRA     BOTH            BOTH ON AND OFF LINE
        STZ     ONSW            NOT ON LINE SWITCH
PROC    SLW     UNIT.           SAVE LOGICAL TAPE NO.
(STH)   LDQ     *+2             LOAD MQ WITH OUTPUT SWITCH + RETURN ADDRESS
        TRA*    $(IOH)          GO TO (IOH)
        TRA     STH
*
BOTH    STL     ONSW            SET ON LINE SWITCH
        CAL     =2B17           MAKE LIKE TAPE 2
        TRA     PROC
        REM
(SCHM)  STL     MONSW.          INDICATE MONITOR CONTROL
        SWT     PUNSW.          IS ON LINE PUNCH SWITCH DOWN
        TRA     (STH3)          NO, WRITE LOGICAL TAPE 3 (PUNCH TAPE)
(SCH)   CLA     MZE2            YES, SET UP TO PUNCH ON LINE ONLY
        LDQ     *+2             ..
        TRA*    $(IOH)          ..
        TRA     SCH             OUTPUT SWITCH AND RETURN ADDRESS
        REM
(STH3)  CAL     =03000000       LOGICAL TAPE NO. FOR PUNCH TAPE
        SLW     UNIT.           INSURE NO ON LINE PRINTING
        LDQ     *+2             SET UP TO WRITE PUNCH TAPE
        TRA*    $(IOH)          ..
        TRA     STH3            OUTPUT SWITCH AND RETURN ADDRESS
        REM
(STHD)  LDQ     *+2             LOAD MQ WITH OUTPUT SWITCH + RETURN ADDRESS
        TRA*    (IOH)           GO TO (IOH)
        TRA     STHD
        REM
```

```
(SPH)   CLA     MZE3            CALL FOR PRINTER ONLY (WITHOUT MONITOR)
        LDQ     *+2             LOAD MQ WITH OUTPUT SWITCH + RETURN ADDRESS
        TRA*    $(IOH)          GO TO (IOH)
        TRA     SPH
        REM
STH3    SXA     STHX,4          SAVE RETURN INDEX TO (IOH)
        LXA     (PUCT),4        UPDATE COUNT OF RECORDS ON PUNCH TAPE
        TXI     *+1,4,1         ..
        SXA     (PUCT),4        ..
        SXD     *+2,4
        LXD     (ELCT),4        ESTIMATED  PUNCHED OUTPUT COUNT
        TXH     TES,4,**        TEST FOR PUNCH COUNT EXCEEDED
        CLA     (PUCT)          HERE WHEN PUNCH COUNT ESTIMATE EXCEEDED
        SSM                     MARK (PUCT) FOR SIGN ON
        STO     (PUCT)          ..
        TSX     $EXIT,4         TERMINATE THIS JOB
        REM
STHD    SXA     STHX,4          SAVE RETURN INDEX TO (IOH)
        STI     SIND.           SAVE INDICATORS
        STZ     MONSW.          INSURE NO ON LINE PRINTING
        LDI     =H              BLANKS
        CAL     1,4
        PDC     0,4
        ADD     =1
        STA     *+2
        TXI     *+1,4,3
        ONT     **,4            CHECK THAT LINE IS NON-ZERO AND NON-BLANK
        TRA     STHD1           OK, WRITE THIS LINE
        TXI     *+1,4,1
        TXH     *-3,4,0
        LDI     SIND.           HERE FOR BLANK OR ZERO LINE
        TRA     STHX            SO SKIP WRITING
STHD1   AXT     1000,4          MAX. LINES OF DEBUG OUTPUT
        LDI     SIND.
        TNX     STHX,4,1        COUNTS DEBUG LINES
        SXA     STHD1,4
        TRA     STH1
        REM
STH     SXA     STHX,4          NORMAL OUTPUT LINE, RETURN FROM (IOH)
        NZT     MONSW.          IS THIS A MONITOR JOB
        TRA     TES             NO, SKIP TO WRITE
STH1    LXA     LNCNT.,4        YES, SO UPDATE TOTAL LINE COUNT
        TXI     *+1,4,1         ..
        SXA     LNCNT.,4        ..
        LXA     (PRCT),4        COUNT PROGRAMMER OUTPUT
        TXI     *+1,4,1         ..
        SXA     (PRCT),4        ..
        SXD     *+2,4
        LXA     (ELCT),4        ESTIMATED PRINTED OUTPUT COUNT
        TXH     TES,4,**        TEST FOR LINE COUNT EXCEEDED
        CLA     (PRCT)          HERE WHEN LINE COUNT ESTIMATE EXCEEDED
        SSM                     MARK (PRCT) FOR SIGN ON
        STO     (PRCT)          ..
        TSX     $EXIT,4         TERMINATE THIS JOB
```

```
        REM
TES     TSX     $(WER),4        CHECK ANY PREVIOUS WRITE
        LXA     STHX,4          RESTORE CALL INDEX
        CAL     1,4             CALL = PZE FIRST,,N
        ARS     18
        ACL     1,4
        STA     MOVE.
        STD     STHC            WORD COUNT INTO OUTPUT COMMAND
        PDX     0,4             AND IR4
        TXI     *+1,4,OUTPUT
        SXA     MOVE.+1,4
        PDX     0,4             RESTORE WORD COUNT
MOVE.   CAL     **,4            MOVE DATA TO OUTPUT BUFFER
        SLW     **,4            ..
        TIX     MOVE.,4,1       ..
        CAL     TES             SET UP ERROR CHECKING
        SLW*    $(TES)          ..
        AXC     STHC,4          ADDRESS OF I/O COMMAND
        PXA     0,4             ..
        STA*    $(WTC)          SAVE IN CASE OF ERROR
        XEC*    $(WRS)          SELECT OUTPUT TAPE
        XEC*    $(RCH)          WRITE OUT THIS RECORD
STHX    AXT     **,4            RESTORE RETURN INDEX
        NZT     MONSW.          IS THIS A MONITOR JOB
        TRA     2,4             NO, RETURN TO (IOH)
        CLA     UNIT.           IS THIS THE MONITOR STACKED OUTPUT TAPE
        SUB     =02000000       ..
        TNZ     2,4             NO, RETURN TO (IOH)
        ZET     ONSW            CHECK TO SEE IF TAPE WAS 100
        TRA     *+3             YES, PRINT ON LINE
        SWT     PRNSW.          IS THE ON LINE PRINT SWITCH ON
        TRA     2,4             NO, RETURN TO (IOH)
        CAL     (PRCT)          YES, PRINT THIS ON LINE
        ADD     =01000000       UPDATE ONLINE PRINT COUNT
        STD     (PRCT)          ..
        TRA     SPH             GO TO ON LINE PRINT ROUTINE
        REM
SCH     NZT     MONSW.          ON LINE PUNCH ROUTINE
        TRA     *+4             SKIP UPDATE OF (PUCT) IF NOT IN MONITOR
        CAL     (PUCT)          OTHERWISE UPDATE (PUCT)
        ADD     =01000000       ..
        STD     (PUCT)          ..
        SXA     NPIR1,1         SAVE IR1
        LDQ     WPUA.           PICK UP ON LINE PUNCH SELECT
        CAL     NPNOP           PICK UP NOP TO AVOID SPACE CONTROL
        AXT     12,1            PICK UP MAX. WORD COUNT FOR ON LINE PUNCH
        TRA     PRPUN.          GO TO BCD TO CARD IMAGE CONVERTER
        REM
SPH     SXA     NPIR1,1         ON LINE PRINT ROUTINE
        LDQ*    1,4             PICK UP FIRST BCD WORD
        PXD
        LGL     6               GET FIRST CHARACTER OF LINE
        PAX     0,1             SAVE IT IN IR1
        CAL     =060            REPLACE WITH A BLANK
```

```
         LGR     6               ..
         STQ*    1,4             ..
         PXA     0,1             FIRST CHARACTER IS CONTROL CHARACTER
         AXT     ESPTB-BSPTB,1
         CAS     ESPTB,1         LOOK FOR THIS CHARACTER IN TABLE
         TRA     *+2             ..
         TRA     SPFND           .. FOUND, GO TO PICK UP SPRA INST.
         TIX     *-3,1,2         ..
         CAL     NPNOP           NOT FOUND, SET FOR SINGLE SPACE
         TRA     SPFND+1         ..
SPFND    CAL     ESPTB+1,1       PICK UP SPRA FOR SPACE CONTROL
         LDQ     WPRA.           PICK UP ON LINE PRINTER SELECT
         AXT     20,1            PICK UP MAX. WORD COUNT FOR ON LINE PRINTER
         REM
PRPUN.   SLW     NPSPR           SET SPACE CONTROL IF ANY
         STQ     WRSA.           SET ON LINE UNIT SELECT
         SXD     TSTCT,1         SET MAX. WORD COUNT
         CAL     1,4             CALL = PZE FIRST,,N
         PDX     0,1             WORD COUNT TO IR1
TSTCT    TXL     *+2,1,**        SKIP IF WORD COUNT OK
         LXD     TSTCT,1         WORD COUNT TOO LARGE, SET TO MAX.
         PXA     0,1
         STA     NPSV4           SAVE WORD COUNT
         ACL     1,4
         STA     NPRC3           FIRST+N
         SXA     NPIR2,2         SAVE IR2
         SXA     NPIR4,4         ..
         LXA     NPSV4,4         RESTORE WORD COUNT
         TXL     1PASS,4,12      IS SECOND PASS NEEDED
         STL     2PSWT           YES, SET SWITCH FOR 2 PASSES
         REM
1PASS    AXT     24,1
         STZ     PBUFF+24,1      CLEAR WORKING STORAGE
         TIX     *-1,1,1         ..
         AXT     1,2             SET FOR LEFT HALF OF CARD IMAGE
NPRC1    CAL     COLIND          INITIALIZE COLUMN MARKER
NPRC2    SLW     PRCOL           ..
         SXA     NPSV4,4         SAVE WORD COUNT
NPRC3    LDQ     **,4            PICK UP FIRST OR NEXT BCD WORD
         AXT     6,4             SET CHARACTER COUNT
NPRC4    PXD
         LGL     6               GET A CHARACTER
         ALS     1               DOUBLE IT
         PAX     0,1             INTO IR1
         CAL     PRCOL
         ARS     6,4             POSITION COLUMN MARKER
         TXL     PDIGIT,1,24     SKIP IF DIGIT ONLY
         TXL     PNZONE,1,95
         TXL     NPRC5,1,96      SKIP IF BLANK
         REM
PNZONE   TXH     PNMIN,1,62      SKIP IF 11 OR 0 ZONE
         ORS     PBUFF+23,2      OR IN THE 12 ZONE
         TIX     PDIGIT,1,32     REMOVE 12 PUNCH
         TRA     NPRC5           SKIP IF + ONLY (NO DIGIT)
```

```
PNMIN   TXH     PNZER,1,94      SKIP IF 0 ZONE
        ORS     PBUFF+21,2      OR IN THE 11 ZONE
        TIX     PDIGIT,1,64     REMOVE 11 ZONE
        TRA     NPRC5           SKIP IN - ONLY (NO DIGIT)
PNZER   ORS     PBUFF+19,2      OR IN THE 0 ZONE
        TXI     PDIGIT,1,-96    REMOVE 0 ZONE
        REM
PDIGIT  TXL     PNDIG,1,18      SKIP IF NORMAL DIGIT
        ORS     PBUFF+3,2       HERE FOR 8-3, 8-4, OR IN THE 8 PUNCH
        TXI     *+1,1,-16       REMOVE THE 8 PUNCH
PNDIG   ORS     PBUFF+19,3      OR DIGIT TO CARD IMAGE
NPRC5   TIX     NPRC4,4,1       COUNTS CHARACTERS
        ARS     1               SET COLUMN MARKER FOR NEXT WORD
NPSV4   AXT     **,4            RESTORE BCD WORD COUNT
        TNX     PNOW,4,1        SKIP TO END IF DONE
        TZE     PNTST           SKIP IF COLUMN MARKER MOVES OUT
        TRA     NPRC2
PNTST   TXL     PNOW,2,0        SKIP TO END WHEN CARD IMAGE COMPLETE
        AXT     0,2             OTHERWIZE SET UP FOR RIGHT HALF
        TRA     NPRC1
        REM
PNOW    TCOA    *               WAIT UNTIL LAST LINE OR CARD IS OUT
        AXT     24,1
        CAL     PBUFF+24,1      MOVE CARD IMAGE TO OUTPUT BUFFER (PBUF1.)
        SLW     PBUF1.+24,1     ..
        TIX     *-2,1,1         ..
WRSA.   WRS     **              SELECT ON LINE I/O UNIT
        RCHA    NPIOC           WRITE THIS LINE OR CARD
NPSPR   PSE     **              SPACE CONTROL IF ANY
        NZT     2PSWT           IS A 2ND PASS NEEDED
        TRA     NPIR1           NO, GO TO EXIT
        STZ     2PSWT           YES, RESET SWITCH
        CAL     PSPR9           SET SPACE CONTROL FOR 2ND HALF
        SLW     NPSPR           ..
        TRA     1PASS           GO THROUGH THE WHOLE MESS AGAIN
        REM
NPIR1   AXT     **,1
NPIR2   AXT     **,2
NPIR4   AXT     **,4
        TRA     2,4             RETURN TO CALLER
        REM
BSPTB   BCI     1,000000
        SPRA    4
        BCI     1,000001
        SPRA    1
        BCI     1,000002
        SPRA    2
        BCI     1,00000+
        SPRA    5
ESPTB   SYN     *
        REM
ONSW    PZE
MONSW.  PZE
2PSWT   PZE
```

```
UNIT.   PZE
SIND.   PZE
PRCOL   PZE
COLIND  MZE
MZE2    MZE       ,,2
MZE3    MZE       ,,3
NPNOP   NOP
PSPR9   SPRA      9
WPRA.   WPRA
WPUA.   WPUA
NPIOC   IOCD      PBUF1.,,24
STHC    IOST      OUTPUT,,**
OUTPUT  BSS       22
PBUF1.  BSS       24
        REM
        COMMON    -176
REC     COMMON    76
PBUFF   COMMON    1
        END
```

```
*       LABEL
*       LIST8
        SUBROUTINE BOARD (ITAPE)
C       PRINTS OUT CHESS BOARD IN READABLE FORMAT.
        DIMENSION FOO(5000), PIECES(43), TAB1(8), TAB2(8), KIND(32),
       1IOCC(64)
        COMMON FOO
        EQUIVALENCE (FOO(2938), PIECES), (FOO(1527), IOCC), (FOO(1463),
       1KIND)
        WRITE OUTPUT TAPE ITAPE,6
6       FORMAT (1H ,18X,5HBLACK/1H ,18X,5H-----)
        DO 1 I = 1,57,8
        DO 10 J = 1, 8
        L = XGETF (J + 57 - 1, IOCC)
        IF (XRANGEF (L, 7, 22)) 7,9,7
 9      IF (KIND(L) - 1) 8,7,8
 8      L = KIND(L) + 5*(XLBITF(L)) + 31
B7      TAB1(J) = PIECES (L+1)
B10     TAB2(J) = SHIFTF(PIECES(L+1), 22)
        WRITE OUTPUT TAPE ITAPE,3
 3      FORMAT (42H **********************************************)
 1      WRITE OUTPUT TAPE ITAPE,4, TAB1, TAB2
4       FORMAT (1H ,8(2H* ,A3),1H*/1H ,8(2H* ,A3),1H*)
        WRITE OUTPUT TAPE ITAPE,3
        WRITE OUTPUT TAPE ITAPE,5
 5      FORMAT (1H ,18X,5HWHITE)
        RETURN
        END
```

```
*       CARDS ROW
*       FAP
        COUNT   20
*       MISTOP
        FUL
        ORG     -11
        IOCD    C,,11
        TCOA    1
        PZE
        REM     MAIN PROGRAM STARTS HERE
C       AXT     *,1
A       CAL     C,1
        ADD     B
D       SLW     C,1
        LGR     37
        TQP     C
        TIX     A,1,1
B       HTR     1
        REM     END OF MAIN PROGRAM
        PZE
        TXI     D,1,C-1
        END
```

```
*       LABEL
*       FAP
        COUNT   35
*       WRITE FOR PRTREE
        ENTRY   WRITE
WRITE   SXD     WRITE-2,4
        CLA*    1,4
        LGR     18
        ALS     15
        LGL     3
        SLW     MOVE
        CLA*    2,4
        LGR     19
        ALS     6
        TQP     *+2
        ADD     =5
        ORA     =H(    00
        SLW     FMT
        CALL    JUNPAK,MOVE,A,B
        TSX     $(SPH),4
        PZE     FMT,,-1
        LDQ     A
        STR
        LDQ     B
        STR
        TSX     $(FIL),4
        LXD     WRITE-2,4
        TRA     3,4
FMT     PZE
        BCI     1,X,2A6)
A
B
MOVE
        END
```

```
*       LABEL
*       FAP
        COUNT   8
*       KEYS SETS AC TO ADDRESS OF KEYS (IN DEC.) AND VARIABLE TO DEC.
        ENTRY   KEYS
KEYS    ENK
        SLQ*    1,4
        LLS     35+18+2
        TRA     2,4
        END
```

10

```
*       LABEL
*       FAP
*BEGIN  INITIALIZING ROUTINE, APR. 19, 1962
        COUNT   88
        ENTRY   BEGIN
        ENTRY   RECOUP
        ENTRY   LDUMP
PMRST   EQU     63
 BEGIN  SXA     DONE,4
        CAL     =6B17
        TSX     $(RWT),4
        CAL     =7B17
        TSX     $(RWT),4
        CALL    FTNBOL
        CALL    STOMAP
        CAL     A
        SLW     PMRST
        STZ     NLOG
        STZ     MLOG
        STZ     IPRINT
        STZ     MOVES
        STZ     NMOVES
        TSX     $TIMLFT,4
        TXH     AC1
        CLA     AC1
        SUB     =900
        STO     AC1
        TSX     $TIMER,4
        TXH     AC1
        TXH     TIMOUT
 DONE   AXT     **,4
        TRA     1,4
TIMOUT  CAL     =100B17
        TSX     $(STH),4
        TSX     TIMFMT
        TSX     $(FIL),4
        CALL    CLOCK,D2
        CALL    PRINT,N
        LAC     6,4
        SXA     PMRST-1,4
        CLA     $(F2PM)
        STA     6
        TRA     $RSTRTN
A       TTR     *+1
        LTM
        SXA     XR4,4
        AXT     FMT,4
 C      SXA     B,4
        STQ     MQ
        SLW     AC1
        ARS     2
        STO     AC2
        CAL     =100B17
        TSX     $(STH),4
```

```
 B       PZE     **,,-1
         TSX     $(FIL),4
         CALL    CLOCK,D2
         CALL    PRINT,N
 XR4     AXT     **,4
         LDQ     MQ
         CLA     AC2
         ALS     2
         ORA     AC1
         TRA*    $(F2PM)
RECOUP   SXA     XR4,4
         LAC     XR4,4
         SXA     PMRST-1,4
         AXT     FMT1,4
         TRA     C
 LDUMP   SXA     XR4,4
         LAC     XR4,4
         SXA     PMRST-1,4
         LXD     LDMPF,4
 LDMPF   TXI     C,,FMT2
N        OCT     777740000000
D2       DEC     2817
AC1
AC2
MQ
TIMFMT   BCI     2,(8H1TIMEOUT)
FMT      BCI     6,(28H1PROGRAM MANUALLY RESTARTED.)
 FMT1    BCI     4,(16H1RECOUP REACHED.)
 FMT2    BCI     4,(15H1LDUMP REACHED.)
         COMMON  12561
 R       COMMON  1
 NLOG    EQU     R+12428
MLOG     EQU     R+9443
IPRINT   EQU     R+3375
NMOVES   EQU     R+3245
MOVES    SYN     R+3246
         END
```

```
*       LABEL
*       FAP
*       FUNCTION LOOK(SQUARE,DIRECTION)
        COUNT   28
*       GIVES FIRST OCCUPIED SQUARE IN GIVEN DIRECTION, OR ZERO.
        ENTRY   LOOK
LOOK    SXA     XR1,1
        SXA     XR1+1,2
        CLA*    2,4
        PDX     ,2
        CLA*    1,4
        SUB     =1B17
        PDX     ,1
LOOP    CLA     IEXTD+1,2               FIND NEXT SQUARE
        ADD     IEXTS,1
        ANA     =020177000000           DO XMOVF
        PDX     ,1
        TXH     NOSQ,1,63               OFF BOARD YET
        ZET     IOCC,1                  SQUARE OCCUPIED
        TRA     FOUND                   YES
        TXL     LOOP,2,8                LOOK AGAIN IF NOT KNIGHT
NOSQ    CLS     =1B17                   PICK UP -1 SO RESULT ZERO
FOUND   ADD     =1B17                   MAKE -0 OR ACTUAL SQUARE
XR1     AXT     **,1                    RESTORE
        AXT     **,2
        TRA     3,4                     RETURN
*       STORAGE ALLOCATION
        COMMON  12561
R       COMMON  1
IEXTS   SYN     R+112o1
IEXTD   SYN     R+11197
IOCC    SYN     R+11035
        END
```

```
*       LABEL
*       FAP
*XTIME    WITH INTERVAL TIMER
        COUNT   15
        ENTRY   XTIME
        ENTRY   XLAPSE
XTIME   TRA     $RSCLCK
XLAPSE  SXA     XIT,4
        CALL    STOPCL,I
        PXD
        LDQ     I
        DVP     =360B17
        XCA
        ALS     18
XIT     AXT     **,4
        TRA     1,4
I       PZE
        END
```

```
*       LABEL
*       FAP
        COUNT   80
*       JUNPAK TRANSLATES MOVES, XFILE GIVES FILES.  FEB 20, 1961
        ENTRY   JUNPAK
        ENTRY   XFILE
XFILE   SUB     =1B17
        ANA     =7B17
        ADD     =1B17
        TRA     1,4
JUNPAK  SXA     XR4,4
        CLA*    1,4
        STO     T1
        TZE     ZERO
        TMI     ZERO
        TSX     $XMV3,4
        PDX     ,4
        ANA     =1B17
        STO     COLOR
        TXL     B,4,6
        TXH     B,4,22
        CLA     KIND+1,4
        SUB     =1B17
        TZE     B
        PDX     ,4
        TXI     B,4,32
B       CAL     PIECES,4
        ANA     =0777777400000
        ARS     12
        ZET     COLOR
        ACL     =H04000U
        ACL     =H0*000U
A       SLW     ANS
        CLA     T1
        TSX     $XMV1,4
        STO     SQUARE
        TSX     XFILE,4
        PDX     ,4
        LDQ     FILES+1,4
        CAL     ANS
        LGL     6
        SLW     ANS
        STQ     ANS2
        CLA     SQUARE
        LDQ     COLOR
        TSX     $XTRANK,4
        ALS     6
        ORS     ANS2
        CLA     T1
        TSX     $XADD,4
        TZE     PKUP
        PDX     ,4
        CAL     PIECES-31,4
        ARS     24
```

```
        ALS     6
        ORA     =H00(00)
        ORA     ANS2
XR4     AXT     **,4
        SLW*    3,4
        CAL     ANS
        SLW*    2,4
        TRA     4,4
ZERO    PDX     ,4
        CAL     SPEC,4
        SLW     ANS
        CAL     =H
        TRA     XR4
PKUP    CAL     =H00
        TRA     XR4-1
T1      SYN     XFILE-2
COLOR   PZE
ANS     PZE
ANS2    PZE
SQUARE  PZE
        BCI     3, SETUP BLACK WHITE
SPEC    BCI     1,REVERT
ZILCH   COMMON  12561
R       COMMON  1
PIECES  SYN     R+9624
KIND    SYN     R+11099
FILES   SYN     R+9519
        END
```

```
*       LABEL
*       FAP
        COUNT   152
*       CHESS ROUTINES IN FAP, RE-ASSEMBLED FOR 709, A. KOTOK
        ENTRY   XLBIT
        ENTRY   XMOV
        ENTRY   XRANK
        ENTRY   XTRANK
        ENTRY   XDEL
        ENTRY   XMV1
        ENTRY   XMV2
        ENTRY   XMV3
        ENTRY   XBAND
        ENTRY   XBOR
        ENTRY   XBEOR
        ENTRY   XBNOT
        ENTRY   STO
        ENTRY   XSTO
        ENTRY   GET
        ENTRY   XGET
        ENTRY   XAND
        ENTRY   XOR
        ENTRY   XLESS
        ENTRY   XNOT
        ENTRY   XONE
        ENTRY   XRANGE
        ENTRY   XADD
        ENTRY   XDEC
        ENTRY   XPRE
        ENTRY   XTAG
XLBIT   LDQ     A1
        STQ     0,4
        TRA     0,4
A1      ANA     =1B17
 XMOV   ANA     M2
        ADD     =1B17
        TRA     1,4
XDEC    LDQ     A33
        TRA     XLBIT+1
XPRE    XCA
        LGL     18
XTAG    ALS     3
        ANA     =7B17
        TRA     1,4
 SUBT   SSM
        ADD     =65B17
        TRA     XRANK
XTRANK  RQL     17
        TQP     SUBT
XRANK   SUB     =1B17
        ARS     3
        ADD     =1B17
A33     ANA     =077777000000
        TRA     1,4
```

```
XDEL    LDQ     A2
        TRA     XLBIT+1
A2      ANA     =01777000000
XMV1    ANA     =63B17
        TRA     XMOV+1
 XMV2   ARS     6
        ANA     =15B17
        TRA     XMOV+1
 XMV3   ARS     10
        ANA     =31B17
        TRA     XMOV+1
XADD    ANA     =077777
        ALS     18
        TRA     1,4
M2              ,,127+8*1024
        REM     XBANDF(L,M) GIVES L AND M
 XBAND  STQ     T1                      M
        ANA     T1                      L*M
        TRA     1,4                     EXIT
        REM     XBORF(L,M) GIVES L-INCLUSIVE-OR-M
 XBOR   STQ     T1                      M
        ORA     T1                      L+M
        TRA     1,4                     EXIT
        REM     XBEOR(L,M) GIVES L-EXCLUSIVE-OR-M
 XBEOR  STQ     T1                      M
        ERA     T1
        TRA     1,4
XBNOT   LDQ     A3
        TRA     XLBIT+1
A3      ERA     =077777000000
T1                              TEMPORARY STORAGE
        REM     STOF AND XSTOF
        REM     STORES X IN A(J) BY CHANGING THE INSTRUCTIONS IN THE PROG
XSTO    BSS     0
STO     STO     T1
        CLA     -1,4
        TPL     LDQ
        REM     PREVIOUS INSTRUCTION WAS AN SXD.  MOVE IT BACK ONE INSTR
        LDQ     -1,4
        CAL     -2,4
        STQ     -2,4
        REM     CHANGE LOC(0,4) TO A STO A+1,4
LDQ     ANA     =077777
        ADD     =1
        ANA     =077777
        ORA     STOR
        SLW     0,4
        REM     CHANGE PPREV INSTRUCTION TO AN LXD -3,4 WHERE J STORED
        CAL     LXD
        SLW     -1,4
        CLA     T1
        TRA     -1,4
  STOR  STO     0,4
LXD     LXD     A,4
```

```
A       SYN     -3
XAND    STQ     T1
        SSP
        ADM     T1
        TRA     1,4
XOR     TZE     1,4
        XCA
        TRA     1,4
XLESS   STQ     T1
        SUB     T1
        TZE     1,4
        CHS
        LRS     0
        PXD
        LGL     1
        ALS     18
        TRA     1,4
XNOT    TZE     NOSAT
        PXD
        TRA     1,4
XONE    SSP
        TZE     1,4
        TRA     NOSAT
XRANGE  TNZ     *+2                     FIX SIGN OF 0
        SSP
        STQ     T1
        CAS     T1
        NOP
        TRA     *+3                     LOWER RANGE IS SATISFIED
NOSAT   CLA     =1B17
        TRA     1,4
        TNZ     *+2                     FIX SIGN OF 0
        SSM
        CAS     -3
        TRA     NOSAT
        NOP
        PXD
        TRA     1,4
        REM     GET AND XGET
        REM     GET  ALLOWS USE OF ILLEGAL SUBSCRIPTS IN FORTRAN
XGET    BSS     0
GET     STO     T1
        CLA     -1,4
        TPL     LDQA
        CLA     -2,4
        LDQ     -1,4
        STQ     -2,4
LDQA    ANA     =O77777
        ADD     =1
        ANA     =O77777
        ORA     CLA
        SLW     0,4
        CLA     PDX
        STO     -1,4
```

```
        CLA     T1
        TRA     -1,4
CLA     CLA     0,4
PDX     PDX     0,4
        END
```

```
*       LABEL
*       LIST8
C       JAN 14, 1961
        FUNCTION ISCHEK(MV)
C
C            THE FUNCTION VALUE IS +1 IF THE MOVE IS A CHECK, 0 IF THE
C            MOVE CANNOT BE A CHECK, AND -1 IF THE MOVE MAY BE A CHECK.
C
C       DIMENSION AND EQUIVALENCE STATEMENTS
        COMMON AA
        DIMENSION AA(4500)
        DIMENSION MAVAIL(100),KIND(32),LOC(32),IOCC(64),NEP(10),MEP1(10),
       1MEP2(10),IBEG(33),IEND(32),LEGAL(5,3)
        DIMENSION IEXTS(64),IEXTD(15)
        EQUIVALENCE (AA(2765),MAVAIL(1)),(AA(2892),K1(1)),
       1(AA(1463),KIND(1)),(AA(1591),LOC(1)),(AA(1527),IOCC(1)),
       2(AA(2900),MCOL(1)),(AA(2914),NUMEP(1)),(AA(2745),MEP1(1)),
       3(AA(2755),MEP2(1)),(AA(1),IBEG(1)),(AA(1495),IEND(1)),
       4(AA(2923),LEGAL(1))
        EQUIVALENCE  (AA(2735),NEP(1))
        EQUIVALENCE (AA(1301),IEXTS),(AA(1365),IEXTD)
        EQUIVALENCE (AA(2903),MOVENO)
C
C            MAIN PROGRAM
 1      M=MV
        MVER=XMV3F(M)
        MVDIR=XMV2F(M)
        MVTO=XMV1F(M)
        MVFR=LOC(MVER)
        MVKIND=KIND(MVER)
        KLOC=XGETF(3-MCOL,LOC)
        IOCC(MVFR)=0
        I8=IOCC(MVTO)
        IOCC(MVTO)=MVER
C            IS THIS AN EN PASSANT MOVE
        IF (NEP(NUMEP)-MOVENO) 8,9,8
 9      IF (XORF(MEP1(NUMEP)-M,MEP2(NUMEP)-M))8,5,8
C            IS THIS A CASTLING MOVE
 8      IF (XANDF(MVKIND-5,XABSF(MVFR-MVTO)-2)) 2,3,2
C            MOVE INVOLVES CASTLING.  FIND ROOK LOCATION
 3      IF (MVDIR-2) 4,5,6
 4      I1=XMOVF(IEXTS(MVTO)+IEXTD(1))
        GO TO 7
 6      I1=XMOVF(IEXTS(MVTO)+IEXTD(3)+IEXTD(5))
 7      I1OCC=IOCC(I1)
        I2=XMOVF(IEXTS(MVFR)+IEXTD(MVDIR))
C            I1=OLD ROOK SQUARE, I2=NEW ROOK SQUARE
C            MOVE PIECES
        IOCC(I2)=I1OCC
        IOCC(I1)=0
C            IS THE KING IN CHECK
        ISCHEK=MABLE(I2,KLOC)
C            RESTORE POSITION OF KING AND ROOK
 12     IOCC(I1)=I1OCC
```

```
      IOCC(I2)=0
      IOCC(MVFR)=MVER
      IOCC(MVTO)=0
      RETURN
C           IS A PAWN PROMOTING
 2    IPROM=0
      IF (XANDF(MVKIND-1,XTRANKF(MVTO,MVER)-8)) 15,16,15
 16   KIND(MVER)=XADDF(M)
      IPROM=1
C           SEE IF MOVER IS CHECKING
 15   ISCHEK=MABLE(MVTO,KLOC)
      IF (ISCHEK) 5,17,21
C           IS THERE A DISCOVERED CHECK
 17   I4=LOOK(KLOC,NORIEN(KLOC,MVFR))
      IF (I4) 5,19,24
 24   I5=IOCC(I4)
      IF (XLBITF(I5-MVER)) 5,25,19
 25   IF (XABSF(KIND(I5)-3)-2) 20,19,20
 20   ISCHEK=MABLE(I4,KLOC)
      GO TO 21
C           MOVE IS NOT A CHECK
 19   ISCHEK=0
      GO TO 21
C           MOVE MAY BE CHECK
 5    ISCHEK=-1
 21   IF (IPROM) 23,23,22
 22   KIND(MVER)=1
 23   IOCC(MVTO)=I8
      IOCC(MVFR)=MVER
      RETURN
      END
```

```
*       LABEL
*       LIST8
C      JAN 14, 1961
       FUNCTION MABLE(MSQ1,MSQ2)
C
C            THE VALUE OF MABLE IS 1 IF THE PIECE AT MSQ1 CAN CAPTURE
C            A PIECE AT MSQ2, AND 0 OTHERWISE.  CHECKS ARE IGNORED.
C
C      DIMENSION AND EQUIVALENCE STATEMENTS
       COMMON AA
       DIMENSION AA(4500)
       DIMENSION MAVAIL(100),KIND(32),LOC(32),IOCC(64),NEP(10),MEP1(10),
      1MEP2(10),IBEG(33),IEND(32),LEGAL(5,3)
       DIMENSION IOPP(16)
       EQUIVALENCE (AA(2765),MAVAIL(1)),(AA(2892),K1(1)),
      1(AA(1463),KIND(1)),(AA(1591),LOC(1)),(AA(1527),IOCC(1)),
      2(AA(2900),MCOL(1)),(AA(2914),NUMEP(1)),(AA(2745),MEP1(1)),
      3(AA(2755),MEP2(1)),(AA(1),IBEG(1)),(AA(1495),IEND(1)),
      4(AA(2923),LEGAL(1))
       EQUIVALENCE  (AA(2735),NEP(1))
       EQUIVALENCE (AA(1285),IOPP)
C
C            MAIN PROGRAM
 1     M1=MSQ1
       M2=MSQ2
       MP=IOCC(M1)
       K=KIND(MP)
       IDIR=NORIEN(M1,M2)
C            CHECK WHETHER PIECES ARE IN LINE
 5     IF (IDIR) 2,2,3
 3     IF (LOOK(M2,IOPP(IDIR))-M1) 2,4,2
C            PIECES ARE IN LINE
 4     IF (K-1) 10,9,10
C            CHECK PAWN DIRECTIONS
 9     I1=IDIR+IDIR-13
       I2=XLBITF(MP)
       IF (XMINOF(XABSF(I1+2)-I2,XABSF(I1-2)-1+I2)) 2,11,2
C            IS THIS A LEGAL MOVE DIRECTION FOR THE GIVEN PIECE
 10    I1=XMINOF((IDIR+3)/4,3)
       MABLE=LEGAL(K-1,I1)
       IF (MABLE) 7,7,6
 6     IF (K-5) 7,8,7
C            PAWNS AND KINGS CAN ONLY MOVE ONE SQUARE
 11    MABLE=1
 8     I2=XABSF(M1-M2)
       IF (XMINOF(I2-1,XABSF(I2-8)-1)) 7,7,2
 2     MABLE=0
 7     RETURN
       END
```

```
*       LABEL
*       FAP
        COUNT   100
*       NORIEN, RECOMPILED FOR 709 A. KOTOK
        REM     FUNCTION NORIEN(MFROM,MTO)
        REM     ROUTINE TO FIND DIRECTION FROM MFROM TO MTO
        ENTRY   NORIEN
NORIEN  CLA*    2,4
        SUB*    1,4                 IS DIRECTION VERTICAL
        TZE     3,4                     EXIT IF FROM=TO
        STO     T1
        ANA     =7B17
        TNZ     NOVT                    IF NOT VERTICAL, TRANSFER
        CLA     T1
        TMI     *+3                     DIRECTION 4
        CLA     =2B17
        TRA     3,4
        CLA     =4B17
        TRA     3,4
NOVT    CLA*    1,4
        SUB     =1B17
        STO     T1
        ANA     =7B17
        STO     VF                      FILE OF FIRST SQUARE
        CLA     T1
        ANA     =56B17
        ARS     3
        STO     HF                      RANK OF 1ST SQUARE
        CLA*    2,4
        SUB     =1B17
        STO     T1
        ANA     =7B17
        STO     VT                      FILE OF 2ND SQUARE
        SUB     VF
        STO     VD                      VERTICAL DIFFERENCE
        CLA     T1
        ANA     =56B17
        ARS     3
        STO     HT                      RANK OF 2ND SQUARE
        SUB     HF
        STO     HD                      HORIZONTAL DIFFERENCE
        TNZ     NOHOR                   DIRECTION NOT HORIZONTAL
        CLA     VD
        TMI     *+3
        CLA     =1B17               DIRECTION 1
        TRA     3,4
        CLA     =3B17               DIRECTION 3
        TRA     3,4
NOHOR   CLA     VD
        SSP
        SBM     HD
        TNZ     NODIG                   NOT DIAGONAL
        PXD                             FIND WHICH DIAGONAL DIRECTION
        LDQ     VD
```

```
        LGL     1
        LDQ     HD
        LGL     1
        SXA     X4,4
        PAX     0,4
        CLA     DIAGD,4          LOOK UP DIRECTION
        TRA     X4
NODIG   CLA     HD               CHECK FOR KNIGHT DIRECTION
        SSP
        ADM     VD
        SUB     =3B17
        TZE     NITE
        PXD                      NO LEGAL MOVE DIRECTION
        TRA     3,4
NITE    LDQ     VD               CHECK WHICH KNIGHT DIRECTION
        LGL     1
        LDQ     HD
        LGL     20
        ADM     VD
        SXA     X4,4
        PDX     0,4
        CLA     NITED,4
X4      AXT     **,4
        TRA     3,4
        REM     STORAGE
                ,,7              DIAGONAL DIRECTIONS
                ,,6
                ,,8
DIAGD           ,,5
                ,,13             KNIGHT DIRECTIONS
                ,,14
                ,,12
                ,,11
                ,,16
                ,,15
                ,,9
                ,,10
NITED   BES
T1
VF
HF
VT
VD
HT
HD
        END
```

```
*       LABEL
*       FAP
        COUNT   248
*       CHESS TABLES FOR COMMON STORAGE
        ABS
A       EQU     1024
        REM     SYMBOL TABLE FOR ARRAYS
R       EQU     -1
 IPDIR  SYN     31284+R
 IOPP   SYN     31278+R
 IEXTS  SYN     31262+R
 IEXTD  SYN     31198+R
 IENUS  SYN     32530+R
 IEND   SYN     31068+R
 IBEG   SYN     32562+R
 JPAWN  SYN     31182+R
 JPROM  SYN     31104+R
 KIND   SYN     31100+R
 M64M1  SYN     31174+R
 MSTO   SYN     31136+R
 MSVN   SYN     31158+R
 NMOV   SYN     31142+R
 KVAL   SYN     29646+R
 FILES  EQU     29519
KCNVAL  EQU     29535
LEGAL   EQU     29639
LOCBEG  EQU     29581
*       LOCBEG AND LEGAL TABLES FOR CHESS.  JAN. 31, 1961
        ORG     LOCBEG-21           LOCBEG TABLE GIVES INITIAL LOCATIONS.
        DEC     56B17,16B17,55B17,15B17,54B17,14B17,53B17,13B17
        DEC     52B17,12B17,51B17,11B17,50B17,10B17,49B17,9B17
        DEC     64B17,8B17,57B17,1B17,61B17,5B17
        ORG     LEGAL-14            LEGAL GIVES LEGAL MOVE DIR. TO MABLE.
        DEC     0,0,0,1B17,0,1B17,1B17,1B17,0,0,1B17,1B17,0,0,1B17
*       CHESS TABLES FILES, LCENSQ, KCNVAL
        ORG     KCNVAL-15
        DEC     8B17,8B17,4B17,4B17,4B17,8B17,8B17,4B17
        DEC     2B17,4B17,4B17,2B17,1B17,1B17,1B17,1B17
        ORG     FILES-7
        BCI     8,KR000QKN0000KB0000 K0000 Q0000QB0000QN0000QR0000
*       KPAWNV TABLE FOR IDVLOP
        REM     MSVN TABLE
        ORG     MSVN-15
C1      BSS     0
        DUP     1,16
                0,0,7*16+7*C1-7**
        REM     IBEG TABLE
        ORG     IBEG-32
                0,0,452         IBEG(33)  IS THE SAME AS IENUS
                0,0,225+112+56+4                  32
                0,0,225+112+4                31
                0,0,225+88                   30
                0,0,285                29
                0,0,257                28
```

```
                0,0,229                 27
                0,0,193+28                  26
                0,0,213                     25
                0,0,205                     24
                0,0,197                     23
                0,0,193                     22
                0,0,189                     21
                0,0,185                     20
                0,0,181                     19
                0,0,177                     18
                0,0,173                     17
                0,0,169                     16
                0,0,165                     15
                0,0,161                     14
                0,0,157                     13
                0,0,153                     12
                0,0,149                     11
                0,0,145                     10
                0,0,141                     9
                0,0,137                     8
                0,0,133                     7
                0,0,105                     6
                0,0,77                      5
                0,0,49                      4
                0,0,21                      3
                0,0,11                      2
                0,0,1                       1
REM             IEND TABLE
ORG             IEND-31
                0,0,452                     32
                0,0,225+171                 31
                0,0,225+115                 30
                0,0,225+87                  29
                0,0,225+59                  28
                0,0,225+31                  27
                0,0,228                     26
                0,0,220                     25
                0,0,212                     24
                0,0,204                     23
                0,0,196                     22
                0,0,192                     21
                0,0,188                     20
                0,0,184                     19
                0,0,180                     18
                0,0,176                     17
                0,0,172                     16
                0,0,168                     15
                0,0,164                     14
                0,0,160                     13
                0,0,156                     12
                0,0,152                     11
                0,0,148                     10
                0,0,144                     9
                0,0,140                     8
```

```
          0,0,136                    7
          0,0,132                    6
          0,0,104                    5
          0,0,76                     4
          0,0,48                     3
          0,0,20                     2
          0,0,10                     1
REM       KIND TABLE M179
ORG       KIND-31
          0,0,6                 QUEEN
          0,0,6
DUP       1,4
          0,0,4                 BISHOP
DUP       1,4
          0,0,3                 KNIGHT
DUP       1,16
          0,0,1                 PAWN
DUP       1,4
          0,0,2                 ROOK
          0,0,5                 KING
          0,0,5
REM       IEXTD TABLE M179
ORG       IEXTD-15
          0,0,2*A+2+15*8
          0,0,A+1+14*8
          0,0,15*A+7+13*8
          0,0,14*A+6+14*8
          0,0,14*A+6+0
          0,0,15*A+7+8              11
          0,0,A+1+2*8               10
          0,0,2*A+2+8               9
          0,0,A+1+15*8              8
          0,0,15*A+7+14*8           7
          0,0,15*A+7                6
          0,0,A+8+1                 5
          0,0,15*8                  4
          0,0,15*A+7+15*8           3
          0,0,8                     2
          0,0,A+1                   1
REM       M64M1      TABLE    M179
ORG       M64M1-15
          0,0,15*64-1
          0,0,14*64-1
          0,0,13*64-1
          0,0,12*64-1
          0,0,11*64-1
          0,0,10*64-1
          0,0,9*64-1
          0,0,8*64-1
          0,0,7*64-1
          0,0,6*64-1
          0,0,5*64-1
          0,0,4*64-1
          0,0,3*64-1
```

```
                0,0,2*64-1
                0,0,1*64-1
        MZE     0,0,1
        REM     IEXTS TABLE
        ORG     IEXTS-63
B       BSS     0
K       SYN     63+B
        DUP     8,8
                0,0,K-*+7*A
                0,0,K-*+6*A
                0,0,K-*+5*A
                0,0,K-*+4*A
                0,0,K-*+3*A
                0,0,K-*+2*A
                0,0,K-*+1*A
                0,0,K-*
        REM      JPAWN TABLE
        ORG     JPAWN-7
                0,0,2           8
                0,0,1           7
                0,0,1           6
                0,0,2           5
                0,0,3           4
                0               3
                0,0,3           2
                0               1
        REM     IPDIR TABLE
        ORG     IPDIR-5
                0,0,2
                0,0,5
                0,0,6
                0,0,4
                0,0,8
                0,0,7
        REM     NMOV TABLE
        REM
        ORG     NMOV-5
                0,0,56
                0,0,8
                0,0,28
                0,0,8
                0,0,28
                0,0,4
        REM     IOPP TABLE
        REM
        ORG     IOPP-15
                0,0,12
                0,0,11
                0,0,10
                0,0,9
                0,0,16
                0,0,15
                0,0,14
                0,0,13
```

```
          0,0,6
          0,0,5
          0,0,8
          0,0,7
          0,0,2
          0,0,1
          0,0,4
          0,0,3
REM        MSTO TABLE
ORG       MSTO-31
DUP       1,32
          0,0,MSTO*1024-**1024
REM       JPROM TABLE
ORG       JPROM-3
          4
          2
          3
          6
ORG       KVAL-5
PZE       0,0,10-1
PZE       0,0,1000
PZE       0,0,3
PZE       0,0,3
PZE       0,0,5
PZE       0,0,1
END       96
```

```
*       LABEL
*       FAP
        COUNT   20
*       FAP
        COUNT   8
        ABS
KPAWNV  EQU     29559
LCENSQ  EQU     29551
        ORG     LCENSQ-15
        DEC     43B17,44B17,45B17,46B17,35B17,36B17,37B17,38B17
        DEC     30B17,29B17,28B17,27B17,22B17,21B17,20B17,19B17
        ORG     KPAWNV-7
        DEC     0,0,4B17,6B17,6B17,4B17,0,0
        END
```

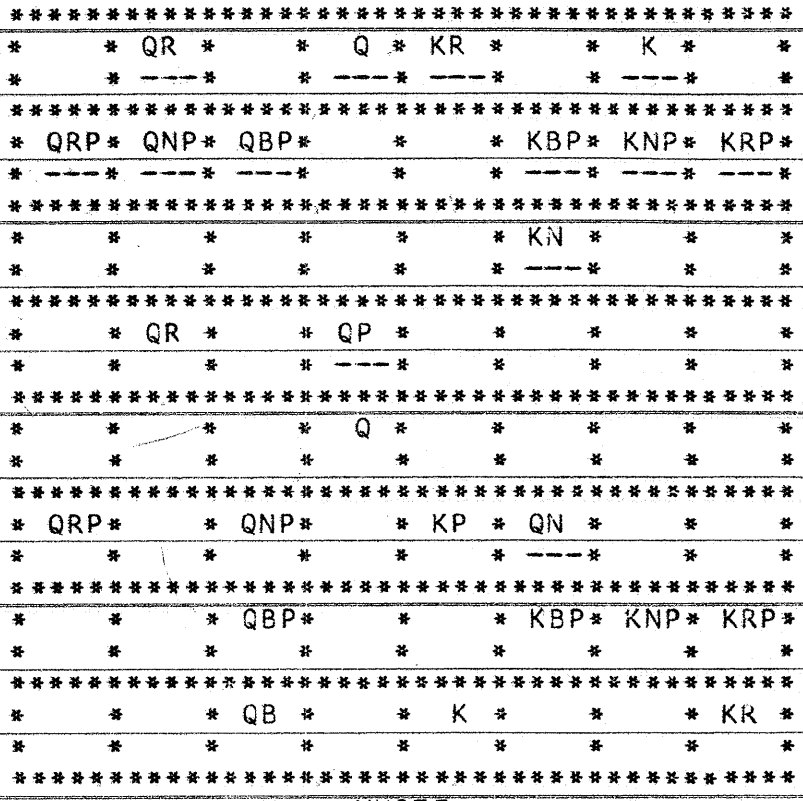
```
*       LABEL
*       FAP
        COUNT   45
*       PIECES TABLE FOR CHESS BOARD PRINTOUT
        ABS
PIECES  EQU     29624
        ORG     PIECES-42
        BCI     1,Q1
        BCI     1,FOO
        BCI     1, B
        BCI     1, N
        BCI     1, R
        BCI     1,Q1 ---
        BCI     1,FOO---
        BCI     1, B ---
        BCI     1, N ---
        BCI     1, R ---
        BCI     1, Q ---
        BCI     1, Q
        BCI     1,KB ---
        BCI     1,KB
        BCI     1,QB ---
        BCI     1,QB
        BCI     1,KN ---
        BCI     1,KN
        BCI     1,QN ---
        BCI     1,QN
        BCI     1,KRP---
        BCI     1,KRP
        BCI     1,KNP---
        BCI     1,KNP
        BCI     1,KBP---
        BCI     1,KBP
        BCI     1,KP ---
        BCI     1,KP
        BCI     1,QP ---
        BCI     1,QP
        BCI     1,QBP---
        BCI     1,QBP
        BCI     1,QNP---
        BCI     1,QNP
        BCI     1,QRP---
        BCI     1,QRP
        BCI     1,KR ---
        BCI     1,KR
        BCI     1,QR ---
        BCI     1,QR
        BCI     1, K ---
        BCI     1, K
        BCI     1,
        END
```

51

TOTAL 4335*

SET OF TABLES NUMBER 30 MOVE IS *QN -KB6

BLACK



WHITE

MAVAIL

K -KB1 K - K2 K - Q1 KNP-KB3

/SAMPLE INITIA INPUT/ 2 QB 1 K 2 KR 2 QBP 2 KBP KNP KRP QRP 1 QNP 1 KP *QN 5 Q 5 QR 1 *QP 9 *KN 2 *QRP *QNP *QBP 2 *KBP *KNP *KRP 1 *QR 1 *Q *KR 1 *K 1.

APPENDIX 2

APPENDIX 3

Record of game played 2/24/62. Machine - white, M. Garber - black

move	White	Black	time
1	KP-K4	KP-K4	1.5 min.
2	QN-QB3	KN-KB3	1.8
3	KN-KB3	QN-QB3	2.2
4	QP-Q4	P×P	4.8
5	N×P	KB-QB4	.8
6	N×N	QNP×N	3.3
7	KP-K5	Q-K2	4.4
8	QB-KB4	QP-Q3	2.2
10	KP×P(Q6)	KB×P(KB7)ch	1.2
11	K-Q2	Q×Qch	.9
12	KB×Q	QBP×P	1.5
13	QB×P	KB-K6ch	2.4
14	K-Q3	QB-QR3ch	1.1
15	QBP-QB4	O-O-O	1.0
16	KB×Nch	K-QN2	.4
17	QN-K4	KB-KB5	.4
18	QN-QB5ch	K-QN3	.9
19	QN-Q7ch	R×N	.9
20	KB×R	B×KB	.3
21	QR-KB1	KBP-KB3	3.4
22	QR-KB5	KR-Q1	3.3
23	KB-K6	KNP-KN3	.6
24	QR-KB1	KBP-KB4	1.5

25	K-Q4	KB-QB4ch	2.2
26	K-QB3	KB-Q5ch	.8
27	K-QN3	K-QB4	.8

average time = 1.8 min./ move

Record of game played 4/21/62. Machine - white

R. Fiorenza - black

move	White	Black	Principal variation		time
1	KP-K4	KP-K4	KP-K4		1.7
2	QN-QB3	QN-QB3	QN-QB3		2.8
3	KN-KB3	KB-QB4	KB-QB4	KB-QN5	2.4
4	KB-QB4	KN-KB3	QP-Q3	QP-Q3	5.4
5	KB-Q5	K-KN1	QP-Q3	K-KN1	6.2
6	K-KN1	QP-Q3	QP-Q3	KB-QB6	5.8
7	KB-QB6	QNP-QB3	QNP-QB3	QP-Q3	3.2
8	QP-Q3	KN-KN5	QB-KN5	QB-KN5	5.5
9	QB-KN5	KBP-KB3	KBP-KB3	QB-KR4	4.0
10	QB-KR4	QP-Q4	QB-QR3	KR-K1	3.8
11	KP-Q5	QNP-Q4	QNP-Q4	KR-K1	4.9
12	QP-Q4	KP-Q5	KP-Q5	KN-Q4	5.1
13	KN-Q4	KB-Q3	KB-Q5	Q-Q4	1.5
14	KBP-KB4	QB-KB4	KB-QB4		3.7
15	KN-KB5	KNP-KN3	KN-KR3	KN-KR6	1.9
16	Q-KN4	QNP-Q5	KB-QB4	K-KR1	.7
17	QN-K4	K-KR1	KB-QN5	QR-Q1	3.6
18	KN-Q4	Q-K1	KB-K2	Q-Q1	4.8
19	Q-KB3	QR-QN1	Q-Q1	QN-Q6	4.3
20	QB-KB6	K-KN1	K-KN1	QB-QN1	2.4
21	QR-QN1	QR-QN5	QN-QN5	KR-K1	5.7
22	QBP-QB3	QR-QB5	QR-QB5	QR-Q1	8.0
23	QR-K1	Q-Q2	Q-QR1	Q-Q1	5.8
24	Q-Q1	QR-QB4	KB-K2	QB-K7	3.9
25	KN-KB3	KR-Q1	Q-K3	QB-K5	8.0

26	QN-QB5	KB-QB4	KB-QB4	K-KR1	4.1
27	KN-Q4	KB-K2	KR-KB1	QB-K5	3.3
28	QR-K7	Q-Q3	Q-QB1		3.2
29	QB-KN5	KR-KB1	KR-Q2	QR-Q7	2.3
30	QBP-QB4	QBP-QB4	Q-Q1		3.5
31	KN-K6	Q-Q8	Q-QR3	KN-KB8	2.0
32	KR-Q1	KR-QB1	KR-KB4	KR-Q7	1.3
33	QR-KN7	K-KR1	K-KR1	KR-Q8	1.0
34	QR-QB7	-	KR-QR1	KR-Q7	3.0

average time = 4.4 min./ move

LEVEL	OPPONENT	MACHINE P-Q4	PRINCIPAL VARIATION						
1	*KN -KB3	QBP-QB4	*KN - K5						
2	*KNP-KN3	QN -QB3	*QP - Q3						
3	*KB -KN2	KP - K4	*QP - Q3	KN -KB3	* K -KN1	KP - K5	*QP - K4	QP - K5	*KN -KN5
4	*QP - Q3	QB -KB4	* K -KN1	KN -KB3	*QN -QB3	KP - K5	*QP - K4	QP - K5	*KN -KN5
5	*QN - Q2	KN -KB3	* K -KN1	KB - Q3	*QBP-QB4	KP - K5	*QP - K4	QP - K5	*KN -KN5
6	*QBP-QB4	KB - Q3	*KN -KR4	QB - K3	*KB - Q5	QB - Q4	*QBP- Q5	KN - Q4	*KP - K4
7	* K -KN1	K -KN1	*KN -KR4	QB - K3	*KB - Q5	QB - Q4	*QBP- Q5	KN - Q4	*KP - K4
8	*QBP- Q5	QN -QN5	*KP - K4	QB -KN5	*QN -QB4	QB -KB6	*KB -KB3		
9	*QN -QB4	KP - K5	*QP - K4	QB - K5	*QB -KB4	KB -KB5	*KNP-KB4	KN - Q4	*KP - K3
10	*KN -KR4	KNP-KN3	*KB - K4	QB - K5	*QP - K4	KN - K5	*KP - K3		
11	*QB -KN5	KR - K1	*KN -KB5	KNP-KB4	*QP - K4	KNP- K5	*QB -KB6	Q -KB3	*KP - K3
12	*KN -KB5	KNP-KB4	*QP - K4	KNP- K5	*QB -KB6	Q -KB3	*KP - K3		
13	*QP - K4	KNP- K5	*QB -KB6	Q -KB3	*KP - K3				
14	*QN - Q2	QN - Q4	*QB -KB6	QN -KB3	*QN -QB4	KB - K4	*KP - K3		
15	*QN - K4	KR - K5	*KB - K4	KB -KB5	*QB -KB6	QN -KB3	* Q - Q8	QR - Q1	*KB -QN7
16	*KB - K4	KB - K4	* Q -QB2	QR -QB1	*KB - Q5	Q - Q4	*QB -KB6	KB -KB3	*KP - K4
17	* K -KR1	KB - K4	* Q -QB2	QR -QB1	*KB - Q5	Q - Q4	*QB -KB6	KB -KB3	*KP - K4